View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Theseus

Bachelor's Thesis (TUAS)

Telecommunication

Information Technology

2010

Jingxiao Zhao

THE WIRELESS APPLICATION AND DEVELOPMENT BUILT ON JAVA 2 PLATFORM, MICRO EDITION

TURUN AMMATTIKORKEAKOULU

TURKU UNIVERSITY OF APPLIED SCIENCES

BACHELOR'S THESIS | ABSTRACT
TURKU UNIVERSITY OF APPLIED SCIENCES
Information Technology | Information Technology
Date: 15-12-2009 Total number of pages: 75
Instructor: Ossi Väänänen
Author: Zhao Jingxiao

THE WIRELESS APPLICATION AND DEVELOPMENT BUILT ON JAVA 2 PLATFORM MICRO EDITION

This thesis is part of a project funded by Jilin University Innovation Fund.
Mobile-computing has become a part of our life today. In recent years, mobile-phone reading has become a new reading styleJ2ME is the platform that SUN has invented for built-in and consumer electronics production. It constitutes the Java core techniques together with J2SE and J2EE.

After the over-platform question has been solved, we plan to design different applied programs for different brands. One wireless application might have hundreds of terminal versions. This became a great handicap for the management of wireless applications update. To solve this problem, we developed several creative methods in the course of inventing a wireless application of the WBOOK (Wireless Book).

The wireless mobile-phone reading book application is different from the general wireless application. Compared with the traditional wireless application, the wireless mobile-phone reading is facing more difficulties caused by the different types of cell phones, that is, one reading program can not suit all the cell phones. Therefore, the present reading programs are just designed for some specific cell phone brands.

The platform can generate a proper program to assure the running of the main program based on different memory sizes of cell phones. As a result, no matter how the content of books is changed, there will be no effect on the other models. The platform offers four options to customers to read the book: WAP (Wireless Application Protocol), Built-in cell phone browser, MMS (Multimedia Messaging Service), and Web download.

This thesis illustrates the developing procedure of MIDP based on J2ME and some referent technology specifically through WBOOK service, WBOOK Server and WBOOK Client. Our aim is that all the WBOOK users can download any eBook at anytime, and enjoy them anywhere. This method has been put into business use.

Keywords: J2ME; value-added; MIDP

Foreword

With the continuous development of wireless communication technology, cell phone as the most popular wireless device is already playing an important role in our lives. The latest research and analysis in wireless applications indicate that

the focus is how the wireless network in the current case of a scalable architecture, strong stability, the terminal automatically matches the value-added wireless data platform.

This thesis describes the development of wireless information technology background, research trends and the J2ME platform, the basic concepts and architecture. Based on the development of J2ME platform for wireless mobile-based value-added data system library (WBOOK) server-side (WBOOK Server) and mobile clients (WBOOK Client), this thesis explains in detail the development process of the MIDP application based on J2ME and related technologies. The focus on the wireless network in the current case is how to structure a scalable, highly reliable, end-value-added wireless data automatically matching system. A prototype system has given the key to the design and implementation. Finally, the thesis evaluates the system briefly, and presents some ideas for further work.

Turku, 26, Nov, 2009

Signature

ZHAO JINGXIAO

Contents

FIGURE LIST

NOTATION

J2EE Java Platform, Enterprise Edition
J2ME Java Platform, Micro Edition or Java ME
WAP Wireless Application Protocol
CLDC Connected Limited Device Configuration
MIDP Mobile Information Device Profile
SMS Short Message Service
BREW Binary Runtime Environment for Wireless
MMS Multimedia Messaging Service
CDMA Code Division Multiple Access
DB Serial Interface D
OOAD Object Orient Analysis & Design
KVM Keyboard、Video(Monitor)、Mouse
JVM Java Virtual Machine

1. Introduction

1.1 Research Background

No one can deny that mobile computing has become part of our basic lifestyle and spread to all areas of people's lives. We can make use of a variety of mobile devices to achieve the journey of information interchange. Because of the popularity and development of mobile computing, our world has accelerated the pace of innovation and integration, which is an unusually broad and varied field. In recent years, using cell phones to read has become a new way of reading among the youths in South Korea and Japan, as well as business personnel. According to Morgan Stanley forecast, by 2008, the market gain of using cell phones to read the wireless value-added would be more than 40 billion U.S. dollars [1]. In 2005, the income of cell phone book for wireless value-added market in South Korean and Japanese was 347 million. However, this market is still blank in China. As different mobile operators take different business models, Japan and South Korea's wireless value-added library system can not directly import into China. Facing a huge market demand, it is urgent to set appropriate conditions for a wireless value-added library system.

However, there are many wireless terminal device manufacturers, and they have different technical standards. These and many other factors have led to a variety of wireless terminal operating system versions, high and low memory capacity, display device sizes and terminal equipment. In addition, the server connection stability is not high, and data transmission and poor process safety problems with the wireless value-added research and development of library systems have created enormous difficulties.

For the wireless terminal operating system, we decide to link Java and wireless devices. One of Java’s features is cross-platform and it could solve the wide range of wireless terminal, especially Sun's J2ME platform has launched a wireless terminal which is currently the most used, and most manufacturers have

also supported platforms. Sun’s J2ME is embedded, and the electronic products launch of the development platform, J2SE and J2EE with Java technology, constitute the three major branches together.

After the Cross-platform issues have been resolved, the application still can not “write once, run anywhere”. Apart from the wireless terminal body and the display device memory constraints, wireless applications have only one or several different brands of wireless terminals to run. The current solution is for a particular memory and display devices of the wireless terminal, to compile different applications. Wireless application procedures may have hundreds of versions for different terminals. This management of wireless applications and upgrades cause great difficulties. On this issue, in the wireless value-added systems in research and development process, we are given a certain innovative solution.

This thesis relates to a wireless mobile data value-added system which has now been put into commercial operation and there are more than 20,000 users on the user terminal every day.

1.2 Research Significance

The best of value-added development services are in South Korea and Japan, based on the book to download to promote the services, and have generally achieved very good economic and social effects. And all kinds of cell phones to read books have become very popular with young people in Japan and South Korea. The main characteristic of this approach is on-demand anytime, anywhere to download e-books. It will be a revolutionary change in reading habits or in the mode of book sales.

However, a wide variety of types of wireless terminal devices result in many difficulties, such as application development and , poor security and updating management. These difficulties affect the wireless applications in the promotion

and popularization seriously. Traditional solutions ways can no longer satisfy the current needs of the market. Therefore, the topic of the thesis has the practical significance and social benefits for the automatic adaptation of wireless terminal equipment and different applications provide terminals for that automatic match.

At the same time, wireless mobile data value-added system books also to some extent, change people's reading habits, are attractive and really easy to read.

Anytime，Anywhere Enjoying Reading，Enjoying Life！

1.3 Research Contents and Objective

According to the requirements of system design and verification, and considering the actual situation of the application for the design and implementation of the platform, I have completed the following major tasks:

- To build scalable value-added wireless books' data system architecture
- To develop a dynamic adaptation feature to fit over 95% of the market cell phone terminal
- To develop mobile client software and set up WAP platform, provides a wide range of terminal access
- To provided billing features and third party library management
- To create a Web mode to download the books on display interface

This thesis describes the J2ME architecture in detail. By designing and developing the Java technology-based, wireless mobile value-added data system library (WBOOK) server-side (WBOOK Server), client (WBOOK Client). It describes the development of wireless applications based on J2ME and related technologies, and proposed dynamic adaptation based on terminal type and terminal.based applications dynamically generate a new development model. At the same time it develops the “Java technology-based wireless mobile

data value-added library system” is a set of value-added wireless data system library.

1.4 Thesis structure

The scope and sequence of the thesis are described as follows:

Chapter I Introduction

It introduces the wireless e-book future development trends and prospects, highlights the development of such wireless applications and the existing problems, and outlines the work of this thesis.

Chapter II Wireless Information Technology Overview

It describes the current mainstream concept of wireless technology; technical characteristics, system structure and research focus of wireless applications research status all over the world.

Chapter III Wireless E-book Systems server-side design

It introduces the server-side design ideas, architecture, and technical problems of this system. It focuses on how the server-side automatically adapts to a variety of equipment as well as a variety of access methods with the server terminal design and implementation.

Chapter IV Design of the wireless e-book client

It describes the client off-line browser, and the on-line browser design concept and implementation.

Chapter V Test Platform

Working out the issues above, the author verifies that the solution is valid, and finally evaluates the system .

.

Chapter VI Summary and Prospects

It summarizes this work and puts forward ideas for projects and further work.

2. Wireless Information Technology Overview

2.1 Introduction of J2ME technology

J2ME, a consumer electronics product development platform, J2SE and J2EE and Java technologies together constitute the three major branches of J2ME technology. J2ME products are divided into two categories: high-end consumer electronics equipment (Connected Device Configuration), the low-end consumer electronics equipment (Connected, Limited Device Configuration).

There are some typical high-end consumer electronic devices: a TV set-top boxes, network video phones, car entertainment / navigation. These devices have rich user interaction capabilities, total memory capacity from about 2mb to 4mb, and they use a continuous, high-bandwidth network connection, usually a TCP / IP connection.

1. In the low-end consumer electronics equipment, there are cell phones, pagers, personal electronic assistants. Such equipment has a very simple user interface, small memory capacity, only a few hundred Kilobytes, and low bandwidth, intermittent network connections. This product are not the network communication but is based on TCP / IP protocol family.
This is the line between two types of equipment with electronic technology and entertainment and technology industries have become increasingly blurred. Bound as a result of manufacturing cost, most of the wireless devices belong to the majority of low-end equipment at present.

J2ME architecture is designed to be modular and scalable so that it can support customers and the embedded device market demands a variety of flexible deployment methods. J2ME architecture defines three basic concepts:

- Configuration: The J2ME configuration is the most concise definition of a public platform. The configuration defines all the similar devices used in Java

language facilities and virtual machine features and the most basic class library.

- Profile: The J2ME profile is a level above configuration (and thus extends this configuration). The profile defines a specifically “vertical” market segment (device family). The main objective of the concept of adding a profile is for specific market segments defined by a standard Java platform, to ensure that the equipment is within the family or the area of interoperability between devices.
- Optional Package: J2ME is a set of API that is set up on the profile. Optional packages include features that are independent of any particular vertical market segment of the device family. The main purpose to design it is to allow this API flexibility in a variety of profile above is loaded. [2]
- The core in J2ME is the CLDC and MIDP standards. They are trying to make Java technology take advantage of resources constraints, and only have limited Internet connectivity to wireless devices. The CLDC and MIDP standard objectives have slight differences, but they complement each other. CLDC attempts to act as a common minimum denominator type of platform used in all types of small-scale, with connectivity devices - independent of any specific device category. MIDP is built-in on top of CLDC, and focuses on a specific device category: wireless, mobile, two-way communications equipment, such as cell phones and two-way pagers. CLDC can also be used to support other types of equipment such as POS (Point of Sale) terminals, barcode scanners, audio-visual equipment, and household appliances.

This article generally refers to wireless devices, such as cell phones, PDAs, as CLDC devices. The features of these devices are weak computer capacity, small memory, limited display device size, wireless connection and so on. At present, one of the largest wireless operators in China mainly develops wireless applications to support this platform.

2.2 WAP Profile

WAP (Wireless Application Protocol) technology is the world's main standard of the mobile terminals to access wireless information services. With the development of mobile communication technology and the Internet technology, WAP technology has become the mobile terminal to access wireless information services with major global standards for achieving the mobile data and value-added services technology base.

The protocol design goal is, based on widely used Internet standards (e.g. HTTP, TCP / IP, SSL, XML, etc.), to provide an air interface and wireless device-independent wireless Internet and comprehensive solution, while supporting the future of open standards. Independent of the air interface refers to the WAP applications (such as voice, fax and E-mail unified messaging, etc.) that can run on top of a variety of wireless bearer networks, such as TDMA, CDMA, GSM, GPRS (General Packet Radio System), CDPD (Cellular Digital Packet Data Network), CSD (circuit-switched data network), SMS (Short Message Service), USSD, etc., without having to consider the question of their differences, and thus maximizing compatibility with existing and future mobile communication systems. Independent of wireless devices means that WAP applications to run from cell phones to powerful PDA and other wireless devices on top of various devices. Manufacturers producing in accordance with WAP should have the same mode of users operation.

The WAP protocol defines a mobile communication terminal connection to the Internet standard and provides a unified, open technology platform, to enable mobile devices to easily access the content in a uniform format of the Internet and Internet information.

The communication model and protocol stack is similar to the traditional WWW communication. The WAP server uses the client depot methods. But there is a

WAP gateway between the client and the server over a WAP model. The client then communicates with the Origin server through the WAP gateway. Meanwhile, the transitional mode between the client and the WAP gateway is also different from the traditional delivery mode between the client and server.

WAP system is mainly composed of three parts.

1. Mobile client (Client): This means the installation of a micro-browser for wireless terminal equipment (such as cell phones) where the WAP pages can be displayed, interpretation, and implementation.

2. WAP Gateway: It is used to complete HTTP protocol to the wireless Internet Transfer Protocol (WSP / WTP) conversion (Protocol Adapters), and wireless Internet content compression (WML Encoder) and compiled (WML script Compiler).

3. Web server: The only difference between the general Internet sites is the web authoring language. The Web server uses the WML (WAP Markup Language) language abbreviations.

WAP content and applications use a similar pattern definition with the WWW. Contents delivery also adopts a set of WWW communication protocol similar to the standard communication protocol. The WAP proxy typically includes two functions:

1. Protocol conversion is responsible for converting WAP protocol stack (WSP, WTP, WTLS and WDP) requesting to WWW protocol stack (HTTP and TCP / IP) requesting.

2. Content encoding and decoding. The encoder is responsible for encoding the content of WAP content into compressed encoding format, thereby reducing Through the use of agent technology, mobile terminal-users can browse a large number of WAP content. At the same time, the WAP proxy allows content and applications to reside in a fixed WWW server,

and using s applications of WWW. The standar client, WAP proxy and WAP servers. Butns. Example, the WAP proxy functionality is included in the WAP server, so that it can acWAP protocol stack takes into account the capacity of the network support, in particular, the capacity to support the cell phone, mobile data services in the early stages of development. The WAPl.X protocol does not directly use the wired Internet HTTP / TLS / TCP protocol, but it uses WSP / WTP / WTLS / WDP protocol, and, at the same time, it increases the use of the WML language. These protocols came out by referencing these fixed protocols (HTTP / TLS / TCP). But it also causes that the cell phones could not access the Internet directly. By the development of the Internet, especially the development of terminals, performance differences in mobile networks and the fixed network transmission decreased. The WAP2.0 protocol implementation is closer to the maturity of the fixed-line protocol (TCP, HTTP). However, in order to ensure support for the existing WAPl.2 compatible cell phones, we must also provide support for the special WAPl.2 protocol stack, so the WAP2.0 dual-stack architecture includes: te WAP2.0 protocol stack. The key feature of WAP2.0 is thl WAP environment. 2.5G and 3G provide more efficient wireless network transfer tocol than WAPl.X.

In short, it is able to provide IP connectivity loading; the WAP2.0 protocol stack takes the WP-TCP instead of WAP1.2 in the WSP / WTP / WDP. Some carriers which can not provide IP connectivity still use the WSP / WTP / WDP co-Instrument Stack Therefore, it also can be said that WAP2.0 returns to the original HTTP / TCP protocol. As far as protocol stack supporting is concerned the WAPl.X is based on the WAP protocol stack. WAP2.0 increased its Internet protocol stack based on common support and services, including TCP TLS and HTTP support. With these two protocol stacks, WAP2.0 can use larger range of network and wireless carriers to provide a connection model. In addition, WAP gateways do not have to implement HTTP / TCP and the WSP / WTP / WDP

protocol conversion; phones do not perform side WSP / WTP / WDP parsing; it is necessary to do HTTP / TCP in wireless configuration. In order to avoid carrying out many protocol conversions, the same hardware devices are used to support more concurrent users. A result of using theWAP2.0 protocol, is that compatible language with the wired Internet is used, which, therefore, achieves a more broad-based support; more applications can more easily be used in mobile Internet. The above mentioned advantages do not apply in the WAPl.X, and this compatibility offers WAP2.0 a longer lifespan and stability of the product than WAPl.X (without the protocol and markup language on frequent upgrades), enabling mobile users to enjoy rich Internet cable content.

2.3 BREW Introduction

BREW (Binary Runtime Environment for Wireless), created by Qualcomm, is a “wireless Internet launch platform,” and value-added services can be developed to run on the basic platform. It provides an efficient, low-cost, scalable, and familiar Application Execution Environment (AEE), focuses on the development and can be seamlessly embedded in any actual handheld device applications. BREW is different in the establishment in the memory space, processor speed. The related hardware has high requirements of the operating systems on top of the high-end products. BREW runs on the type of existing equipments. Now, the BREW environment has the same functionality provided by the operating system on a PC, and it is possible to download specified types of applications or games of the service providers. At the same time, through the BREW interface functions, the supplier can provide a complete package of information, business and entertainment functions. In a future version, BREW core classes will be able to provide Bluetooth technology, Global Positioning System (GPS) and data-based business phone services.

BREW provides a set of Application Program Interfaces (API). The manufacturers and developers can extend the operating environment at any

time; provide a variety of applications require additional performance modules, such as “wireless Internet launch platform” contained in the multimedia, multi - connectivity, location-based services, user interface, network and other functions suite.

The disadvantage of BREW platform is that, the current BREW development tools are not mature, primarily using C for developing it. In addition, there are 34 operators using Java all over the world, while only eight operators use BREW, so it has a small scope of application.

3. Wireless E-book Systems server-side design

3.1 Functions and demands

It is a huge challenge to create a framework for a wireless electronic book system which supplies the service and the terminal to read the required client. One of the difficulties is there are many brands of wireless terminals. The current developer re-developed a suitable version manually for different terminal devices. This method could deal with one-to-many in the wireless business and entertainment applications, but for a few tens of thousands of books, the many-to-many method is basically impossible with the traditional development mode. In this system design, our team adopted a “terminal automatically adapts” approach to solve this problem On the other hand, as the mobile operators use different platform techniques, the client must support two kinds of common platform requirements: the mobile data value-added platform requirements (J2ME), and Unicom's value-added data platform requirements (BREW). The server must provide all different types of data acquisition methods that the client needs. Including e-book on-demand services, MMS, WAP e-book on-demand / download service, common platform for online services, a common platform for offline reading packages automatically generates and downloads services. These business need to complete all the service requests with the uniform contents. In the traditional design and development of wireless applications, the data and procedures are not isolated but this does not apply here. We need a new approach. As WBOOK platforms need to provide a variety of ways to read books, the contents will be packaged into a variety of data forms. It is important to deal with the contents of books. We save the books’ information in a database. Each module automatically assembles the electronic books to achieve that, the users could enter one time, but use multi-time. The system architecture must include books content entry, copyrights management and user management, system manage subsystem and other modules. These modules are relatively simple, and are, therefore, covered only briefly below.

Here are the mentioned basic needs of WAP, Web, and MMS. The client software, due to both the off-line and on-line versions of the demand, will be introduced in detail in the fourth chapter.

3.1.1 WAP Demands

Customers use cell phones via WAP to access the WBOOK platform that can perform the following actions:

- Online search required books (It supports a variety of ways to search, including the author, book name, book type, main character name and so on, it also supports fuzzy queries, and list queries)
- Selection of books to read online
- On-line download models fit their own off-line reading packet
- On-line download models fit their own on-line reader
- Cost of inquiries, as well as payment service.

3.1.2 Web Demands

Customers through the web client can access the WBOOK platform that can perform the following actions:

- Book a more user-friendly interface to search
- On-line download models to fit their own off-line reader
- On-line download cell phone models to fit their own online reader
- WAP download via the selected books was required ID
- The cost of inquiries, as well as online bank card payment service
- Custom MMS e-book service

3.1.3 MMS needs

After the multimedia message e-book service has been customized, the following services can be received:

- Access to the SMS subscription services, unsubscribe services, query the cost

- Mode selection of books via SMS
- Receiving and reading the selected books with MMS(Multimedia Messaging Service).
- Other value-added services

3.1.4 CS USER Demands

Customer Service uses the user management system to perform the following actions:

- View the user's personal information; web on-line, WAP on-line, reader connection status.
- View the user's payment of premiums, orders
- View the cost of books
- Alter the cost of books, by request
- Alter of the cost of the user's request

3.1.5 CS Upload Demands

Customer Service uses books management system to do the following:

- See all the book information
- Add new book information
- Modify their own information on the book entry
- Request the removal of the own book input from the information input.

3.1.6 System Management Requirements

The System management

- Records the operation of customer service use in the user management subsystem
- Records the operation of customer service use in the books management subsystem
- Records the operation of management staff using the system
- Records the customer web access to the logs
- Records the customer WAP access records

- Records the customer search keywords
- Records all the anomalies that occur

3.2 System design and structure

The system uses the J2EE framework to construct a system, which uses the standard J2EE containers (Web container and EJB containers) to complete the searching, registration, wake-up call and destruction of the WBOOK components Client software and the WBOOK server communicate through the HTTP protocol; the Web server and the WBOOK application servers communicate via RMI. The WBOOK components connect to the server through the spatial data to communicate with the map data server, and then complete the database access. The integration of legacy systems is completed by the Message Service (JMS) infrastructure.

As shown below, the system is composed mainly by the client-side software, Web server, WBOOK application servers, WAP servers and data servers. The client software is responsible for issuing service requests to the WBOOK platform and receiving the return data of the WBOOK platform. Based on HTTP, the Web server performs session management, status management, log management and other functions. The WBOOK application server is an application based on J2EE. As a service running in the background, it is composed of the WBOOK Servlet engine, session components and entity components. It is used to complete WBOOK business logic, including automatic terminal pairing, a variety of data storage, data query, and user personalization analysis and other services.

Server-side WBOOK session component (Servlet): The WBOOK session components behave as a single WBOOK object executed by a client; as a WBOOK entity component object client, the WBOOK session component can access a number of different types of WBOOK entity components to complete a

session. The prime responsibility is communication with the client (hand-held devices, Web client, WAP client), and parsing incoming requests, and then sending them to the WBOOK entity components; receiving the response of the WBOOK physical components, and generating client-side responses. The server-side WBOOK entity components (EJB): parsing and generation services, the implementation of spatial data query logic, according to customer's WBOOK request, generate a collection of specific information and other services. Multiple clients can share at the same time, and access the same WBOOK entity component. Through the transaction of accessing or updating the underlying data, data integrity can be guaranteed.

In system-level architecture for distributed modular architecture, the overall level is divided into three levels: interface (boundary), control, and entity. Each subsystem has its own implementation of these three categories. In order to have a good scalability, that is, the expansion of a complete sub-module, the interface of each module can be distributed to the other.

The overall structure of the system is shown as follows in a case diagram:

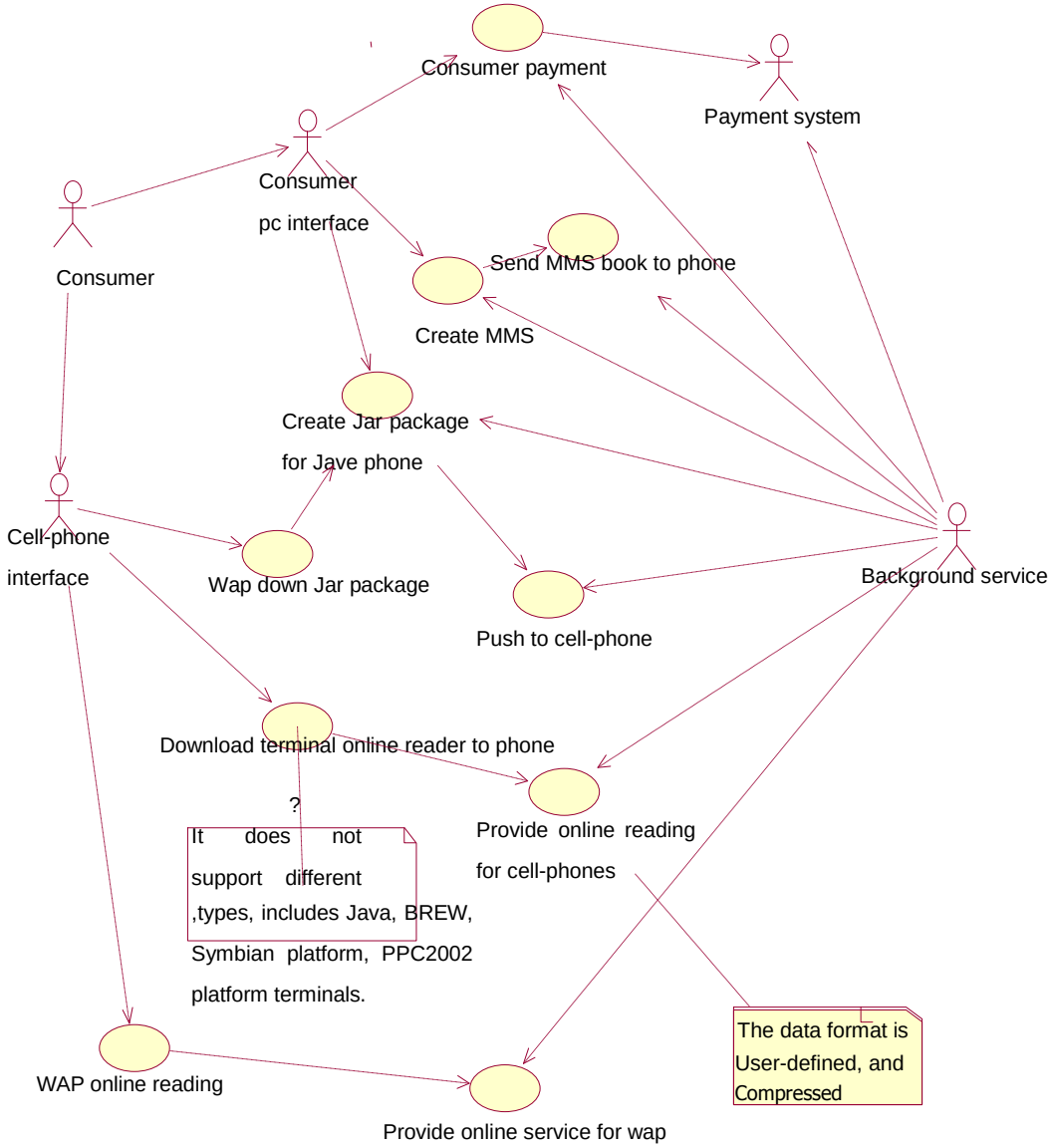


Figure 1 System use case diagram

As shown in the diagram, the entire system is divided into several sub-systems to achieve these use cases of the rules. The next figure is the sub-structure diagram:

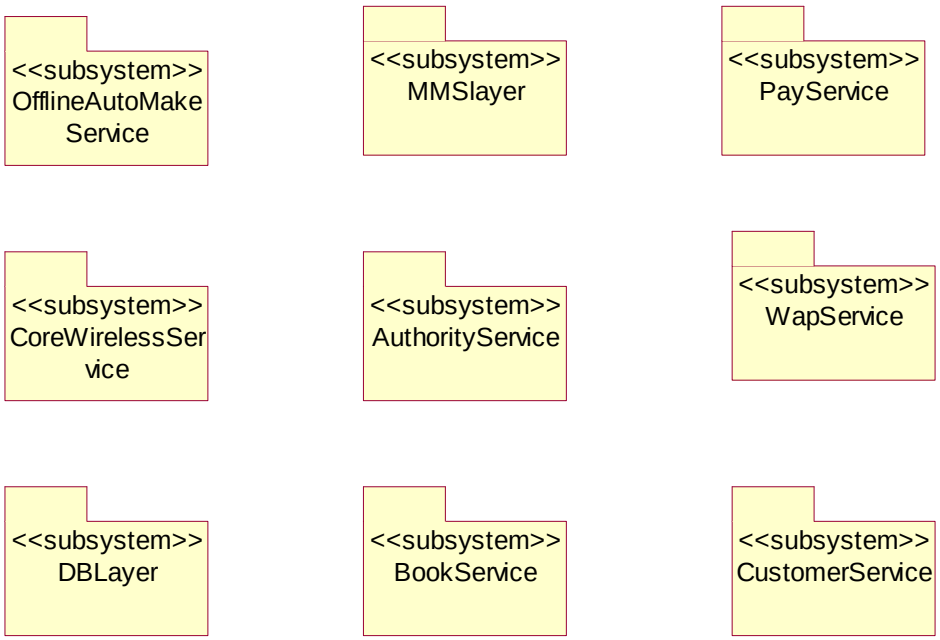


Figure 2 System diagram

The system is divided into nine sub-systems:

1. The Off-line Auto Make Service is automatically generated by off-line terminal subsystem,

2. The MMS Layer is a multimedia e-book generation subsystem,

3. The Pay Service is the online payment subsystem,

4. The CoreWirelessService is the client online subsystem,

5. The Authority Service is to empower the sub-system,

6. The WapService: Wap downloads and online services is a subsystem,

7. The DB Layer is the database layer; each subsystem is responsible for providing a unified database interface, and providing an expanded base class.

8. The WBOOK Service is a book management subsystem.

9. The Customer Service is a customer service subsystem.

These 9 sub-systems follow the object-oriented design principles.

The design tool is Rose2003 which is created by Rational, so there are many class diagrams, and timing diagrams to carry out the detailed design, as can be seen from the figure illustrating the relationship between the various classes, as well as the principles.

3.2.1 Off-line terminal automatically generated subsystem

This sub-system is the core platform, which enables automatic adaptation to different cell phone functions. Different models of cell phones support Java differently, and cell phone memory also limits the executable packet with the extension is “.jar” size. For example, Nokia 7600 supports a maximum of 64k, of jar package while the Motorola A760 supports up to 512k of the jar package. The off-line terminal automatically generates a system to collect the front-office and cell phone models, find a matching Java support type from the database, and a right-to-read automatic generation e-books for cell phones.

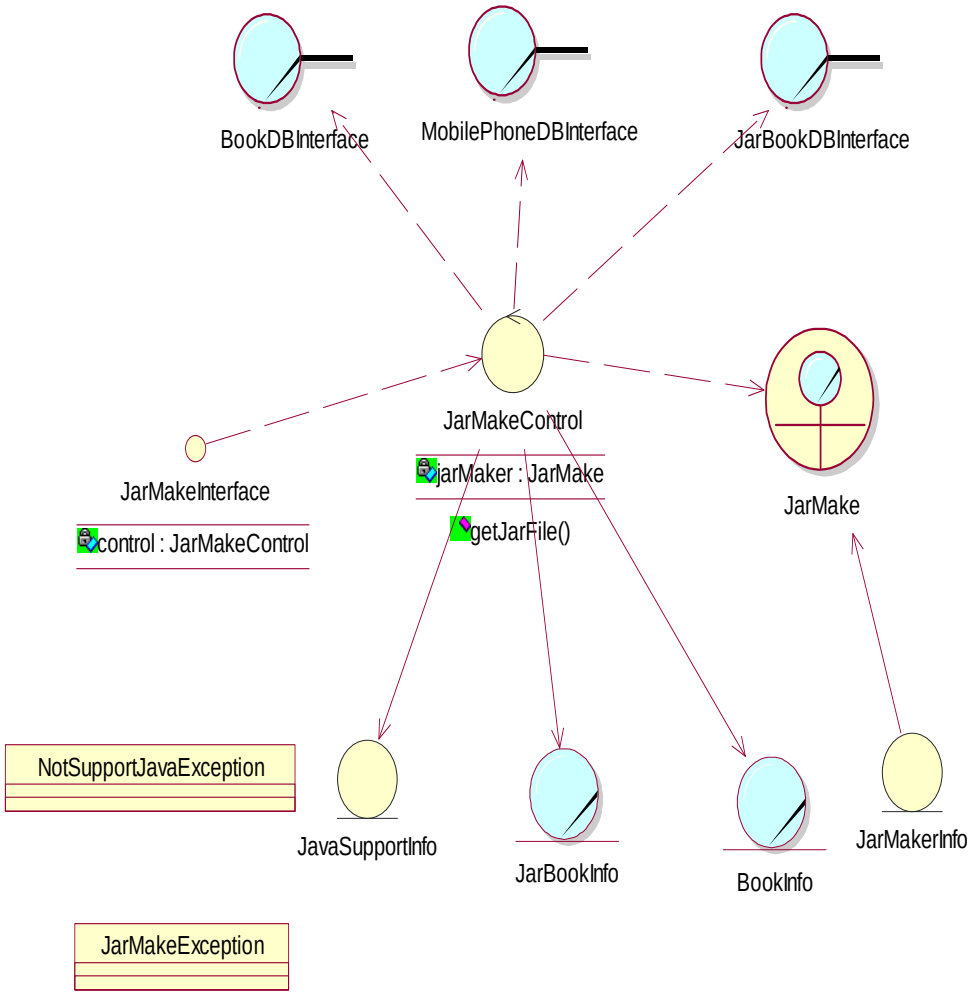


Figure 3 The main design

From the design figure, we can see that the core control class is JarMakeControl, which is responsible for automation. There are three database interface layers thrown to the outside interface class is JarMakeInterface and four entity classes. JarMake is the auxiliary sub-category of control class JarMakeControl; it is the work class of the actual control auto-generated off-line terminal package.

The following diagram gives a concrete realization:

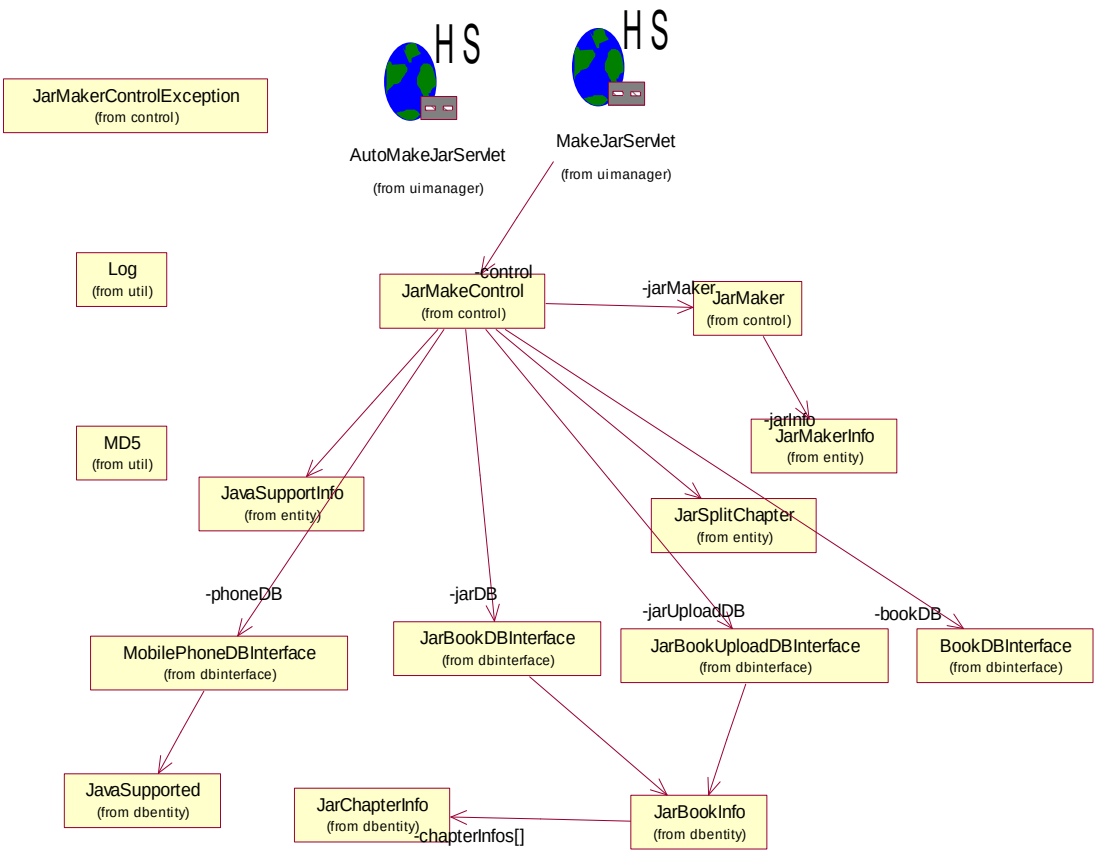


Figure 4 Implementation class diagrams

We skip the detailed logic of the UI layer, and used two edge interfaces of interface layer to represent the UI needs. Here the figure records the system log and error log carried out by Log class. Log is a lightweight logging of tool classes; it is used for various sub-systems under the util package. JarMaker actually packages, and the entity class is JarMakerInfo. JarMakerInfo is necessary for packing ageing information, while the other four entity classes are used for logic control.

3.2.2 Database sub-system

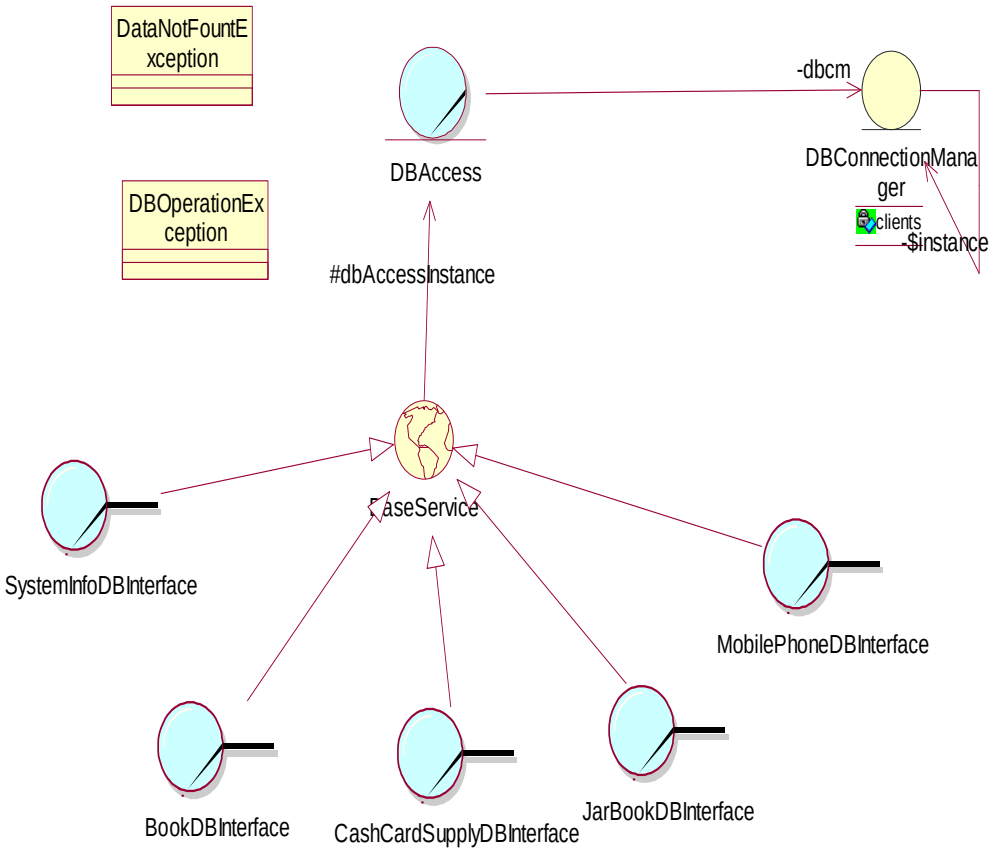


Figure 5 Database sub-system logic

Figure 5 above shows the design logic of the database layer. It has a DB (Serial Interface) Access instance for the outside world inherited base class—BaseService. DB Access encapsulates common operations for the database, including the implementation queries, change, delete, etc., and it provides an interface to prepared statements. The Class of DB Connection Manager is responsible for establishing the connection with the database, including a buffer pool. When a new connection is requested, it obtains a connection from the built buffer pool. If it is not free, it then re-creates a new connection. In this way, other modules just need to inherit BaseService, and can then automatically initialize the connection with the database, use a variety of examples of DBAccess interfaces without having to re-implement the

database-related detail operations, such as reconnection, etc. This thesis deals with the logical operation.

Here is the implementation class diagram:

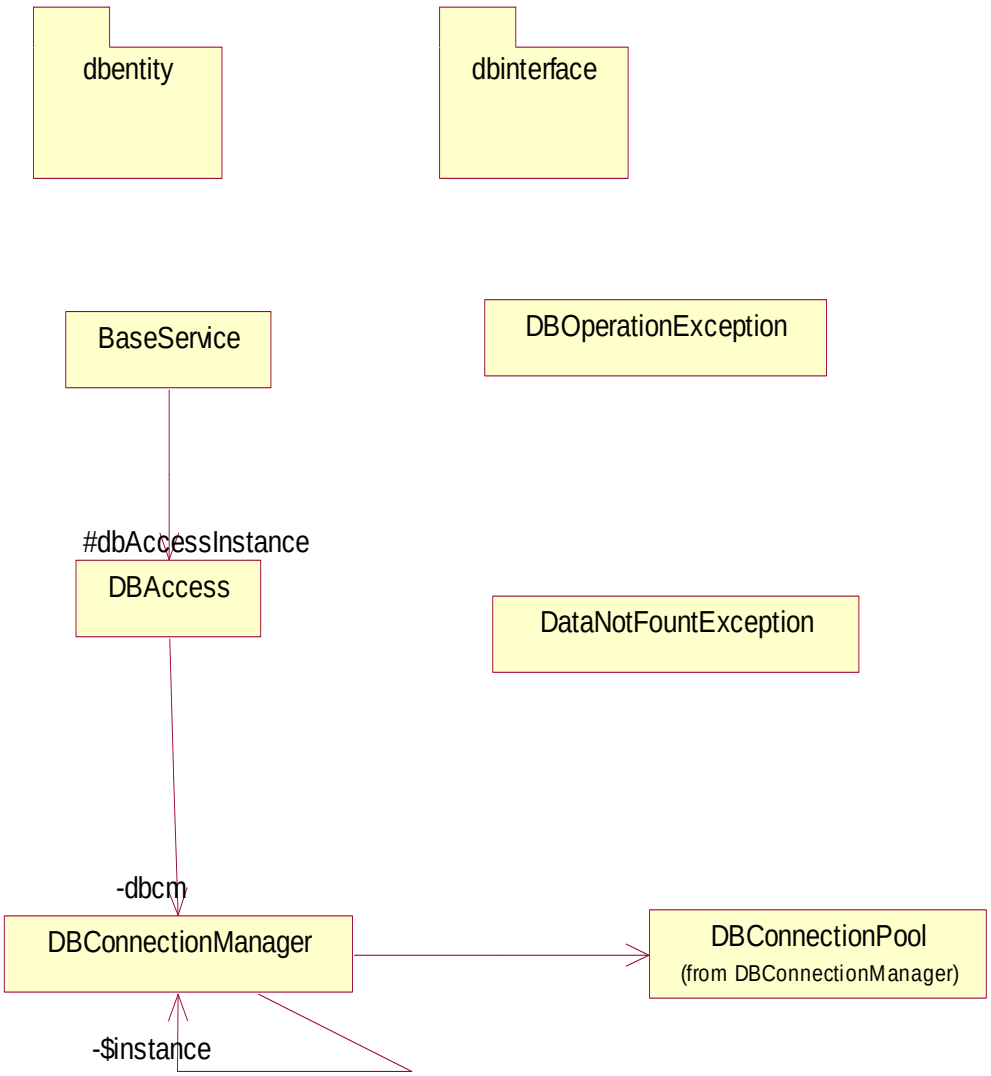


Figure 6 Implementation class diagrams

Figure 6 shows that, in the operations of database, two kinds of exceptions may occur: DBOperationException and DataNotFountException whereas the other modules just need to create a subclass of BaseService, and packaging-related logical operations. Its subclasses can choose to be placed in DB interface Package. The package can also be self-contained with its modules, which are used in whichever the entity class the db entity package is placed or it can be

self-contained with its modules. So it is consistent with the principles of interface segregation.

3.2.3 Payment subsystem

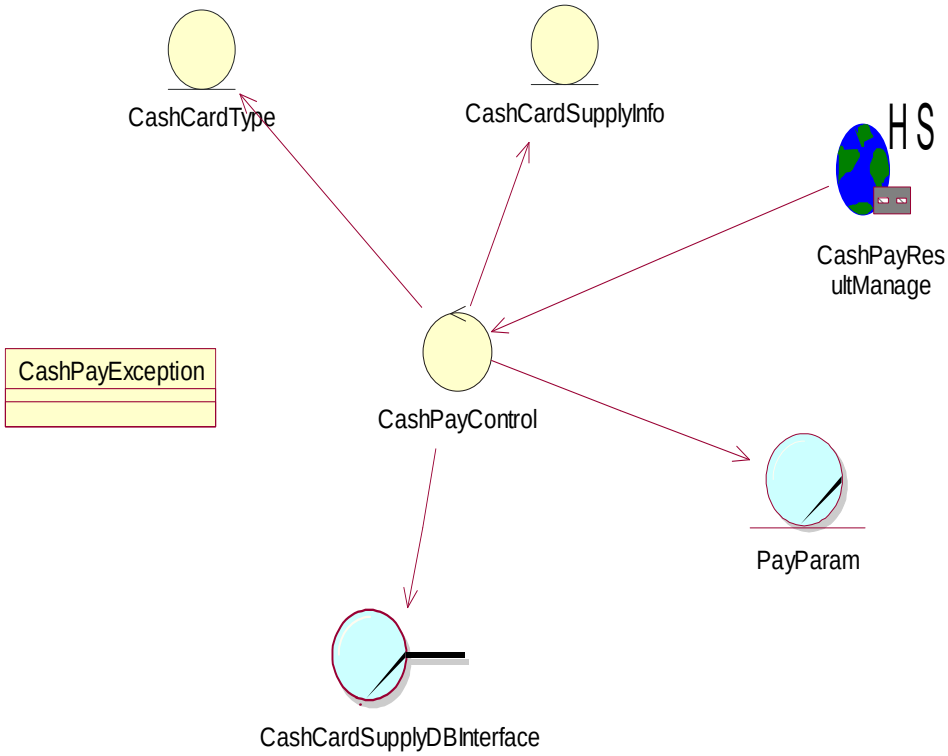


Figure 7 Payment Sub-system

The payment sub-system design figure shows the business logic sub-systems, including control class which is CashPayControl, entity class which is CashCardType and CashCardSupplyInfo.The database interface class is CashCardSupplyDBInterface: the exception class is CashPayException, the edge interface class is CashPayResultManage, and parameter type is PayParam. The outside world contacts the whole payment subsystem with the edge class.

Here is implementation class diagram:

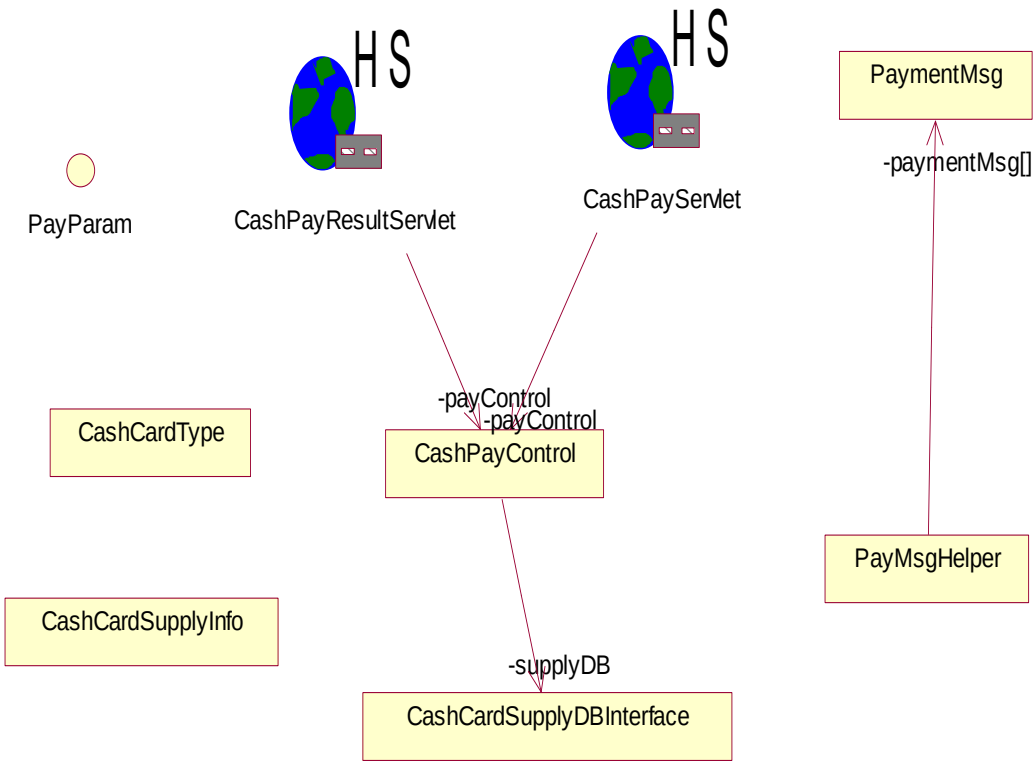


Figure 8 Implementation class diagrams

From the above figure we can see that there is a CashPayServlet acting as a competing interface classIts auxiliary class is PaymentMsg and PayMsgHelper.

3.2.4 Management Subsystem original book

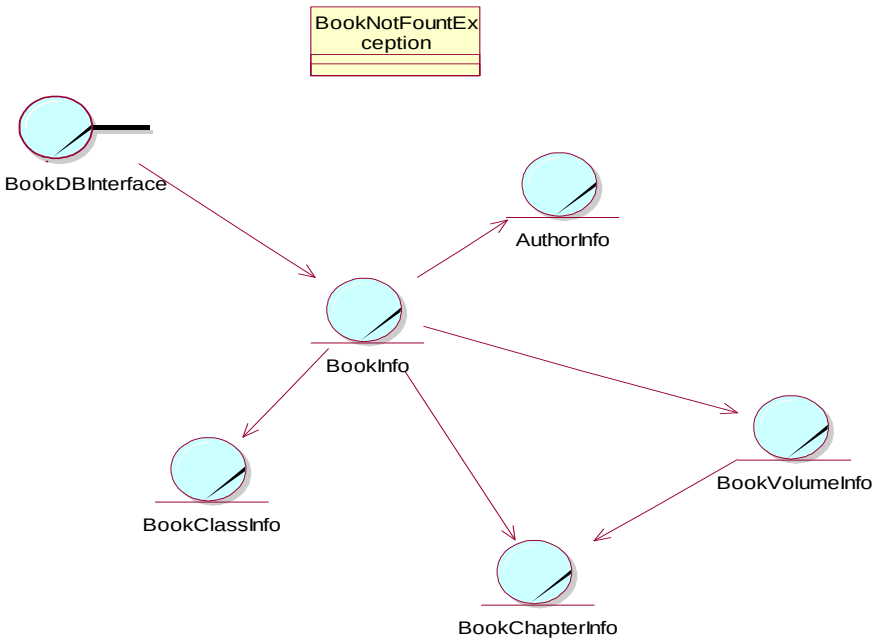


Figure 9 Books, original management sub-system

From Figure 9 it can be seen that there are five entity classes. They provide the external interface class, that is, the edge type which is BookDBInterface. The entity classes here are based on the database level of service. The actual control class inherits the interface class, and the realization of all interfaces.

Here is the specific implementation class diagram:

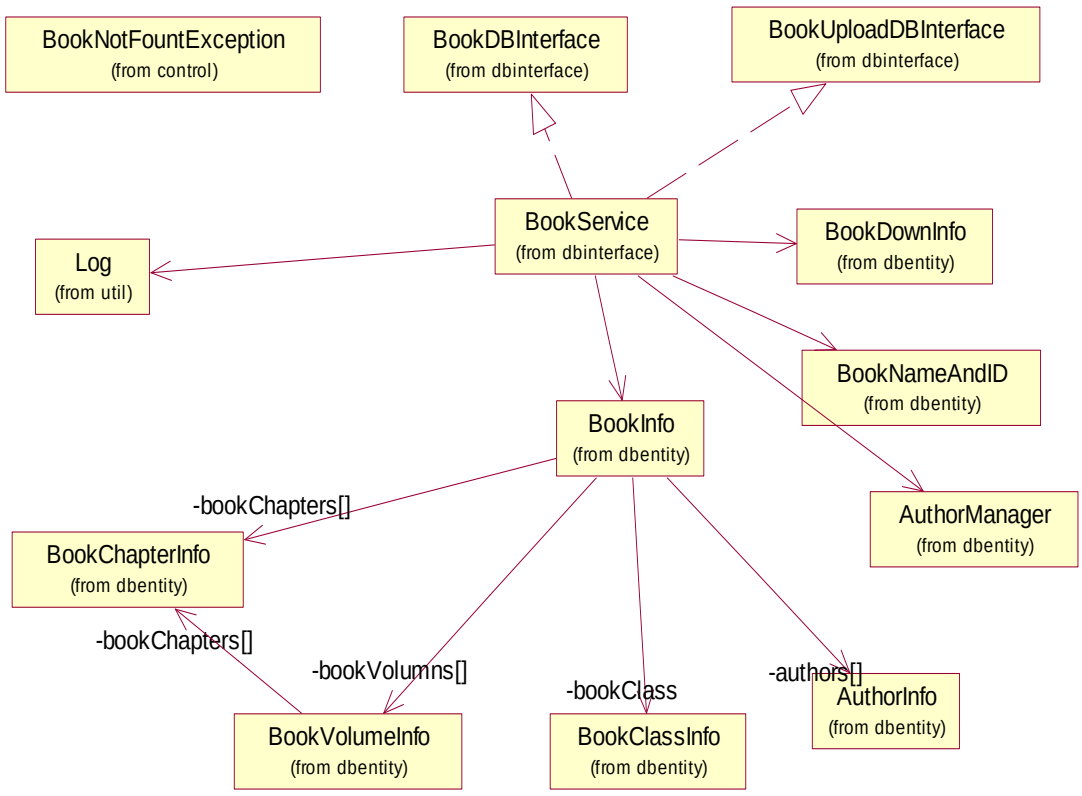


Figure 10 Implementation class diagrams

From Figure 9, we can see that the core control class is BookService and the two edge interface types are BookDBInterface and BookUploadDBInterface. The control class implements interface class for all interfaces, external calls two interface classes to perform the task. There are seven entity classes used to control the class. The Log of this sub-system is still recording all kinds of information.

The detailed class diagram information is no longer listed.

3.2.5 Hand-held Terminal Online Service

The services are divided into three types:

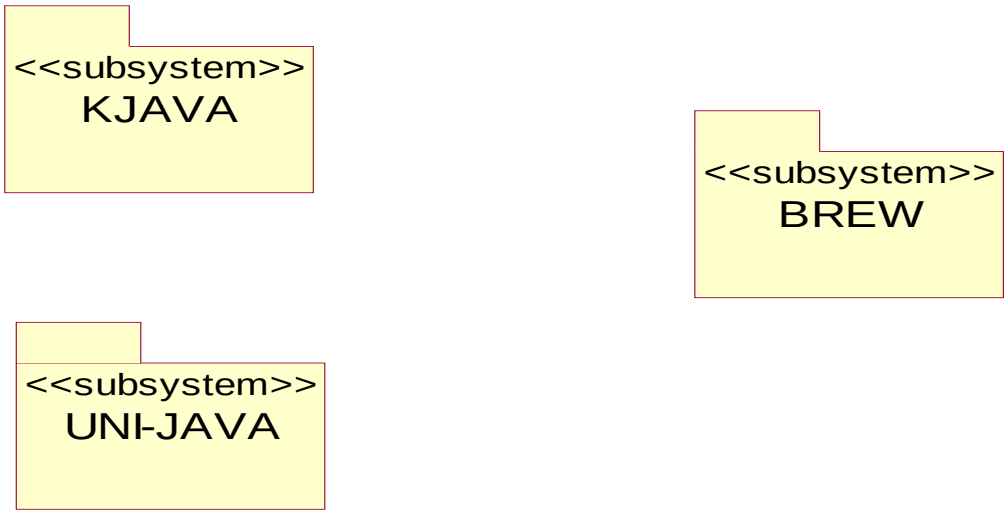


Figure 11 Three types of service

The three types of design principles and structures are similar, here we highlight the KJava design, and its design as follows:

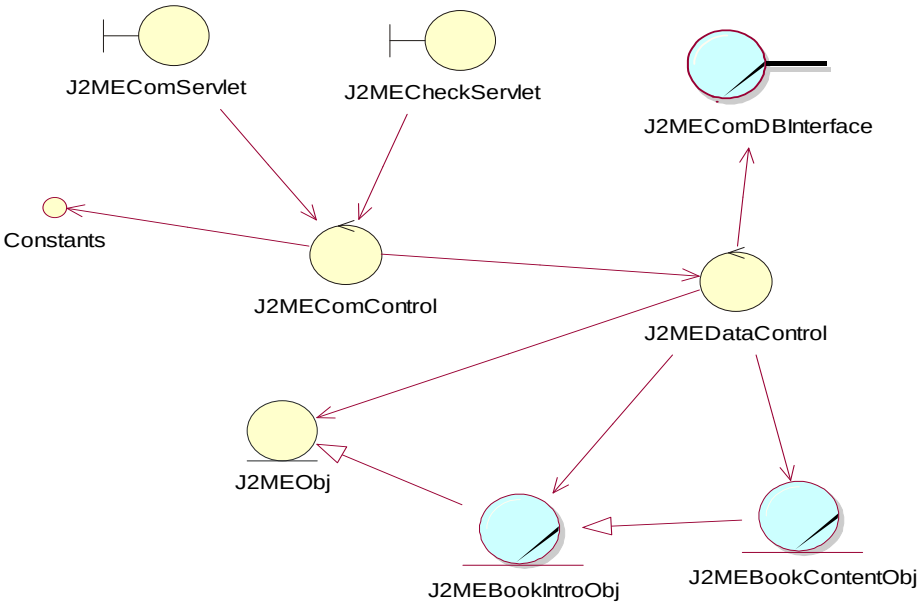


Figure 12 KJava design

As can be seen from Figure 12, the two interface classes; J2MECom and J2MECheck, are respectively used for exchanging and testing data. The J2MEComDBInterface is the database interface. There are two control classes; the Core control class is J2MEComControl and the data control class is J2MEDataControl. The data control class is mainly used for data management and cache management, while the logical implementation is achieved by J2MEComControl which has three entities classes.

The above figures demonstrate that each subsystem has its own control class, interface class, and the entity class, which is the basic requirement for OOAD (Object Orient Analysis & Design).

3.3 The procedures for dealing with process design

Here we use the timing diagram to illustrate the procedure of a process design. Due to the complexity of the system, we will first explain a number of core processes.

3.3.1 Off-line terminal automatically generated subsystem

Here is a sequence diagram from a user requesting after DownloadAction.

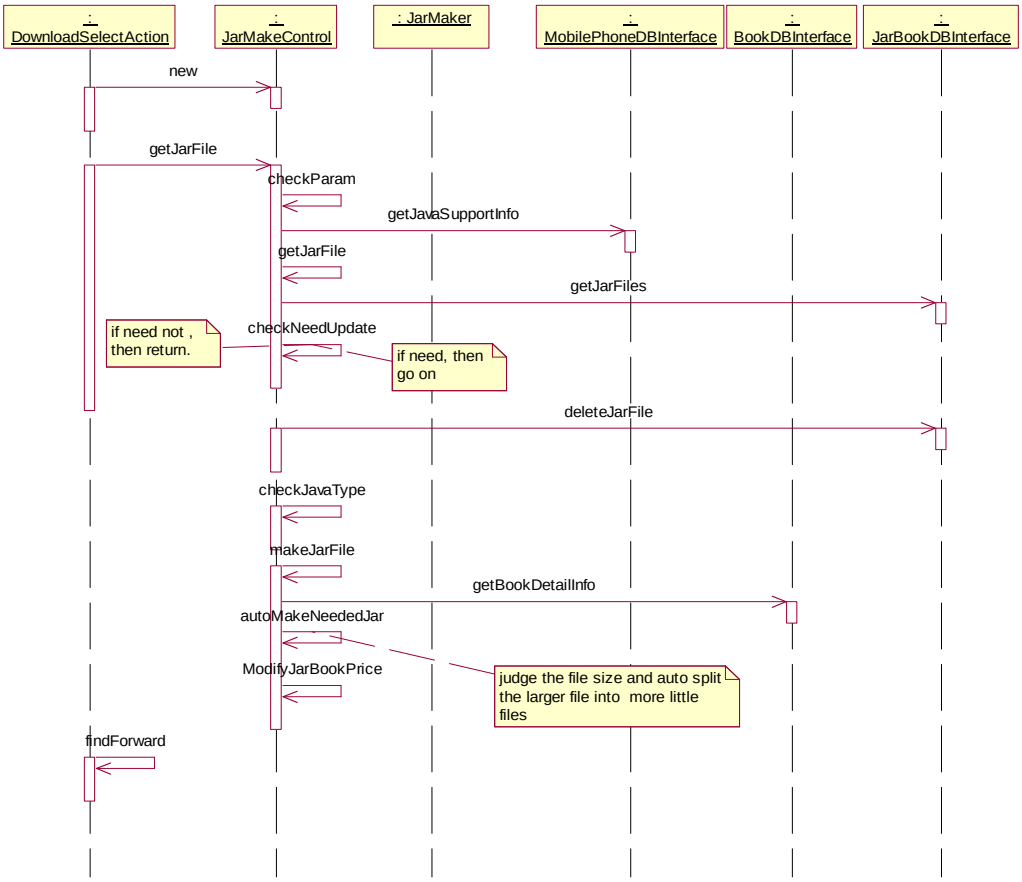


Figure 13 The sequence diagram after DownloadAction

When outside action generates the request, the system will set an instance of a JarMakeControl object, and then obtain the generated objects by getJarFile interface to, finally return to the end-users.

Here when the calling interface is generated, the control class firstly determines whether the phone model has been generated over the same book. If the phone model has been generated before, and does not require replacement, it directly goes back to this book. If it needs to be updated, it then deletes the previous records. Then it deals as “no” and it continues to generate the book package. First, it checks the types of phone's java supports, and then it obtains the books contents bases’ ID. After that, it uses JarMaker to generate the final package. Before the generation, it needs to split the large packet into a suitable size

packages, and then the record is saved to the database. Finally, the record will be back to the latest results package.

3.3.2 -Java On-line Service

The on-line services need to verify the client version information and are divided into two cases. One is while the client is visiting, that is, accessing the information version and then client-side validation. The other situation is when the client request with version information is validated by the server-side. The system uses both of them for convenience. Every time the client enters the procedure, first it processes the validation, and then the certificates on the specified interface.

Here is the first verification timing diagram:

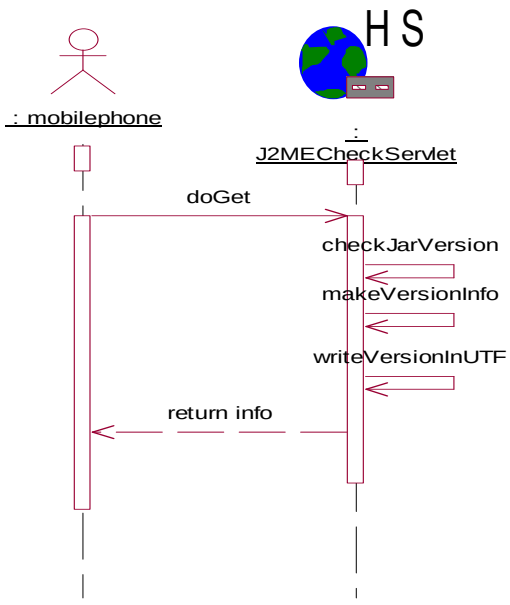


Figure 14 The first timing diagram verification

On-line services need to provide a user name and password. It is convenient for statistics and management for the user. The password here uses MD5

encryption for transmission. Due to the irreversibility of the MD5 encryption algorithm, it ensures that the user's private information is not leaked.

The following diagram illustrates the online client access to the root directory of the on-line timing:

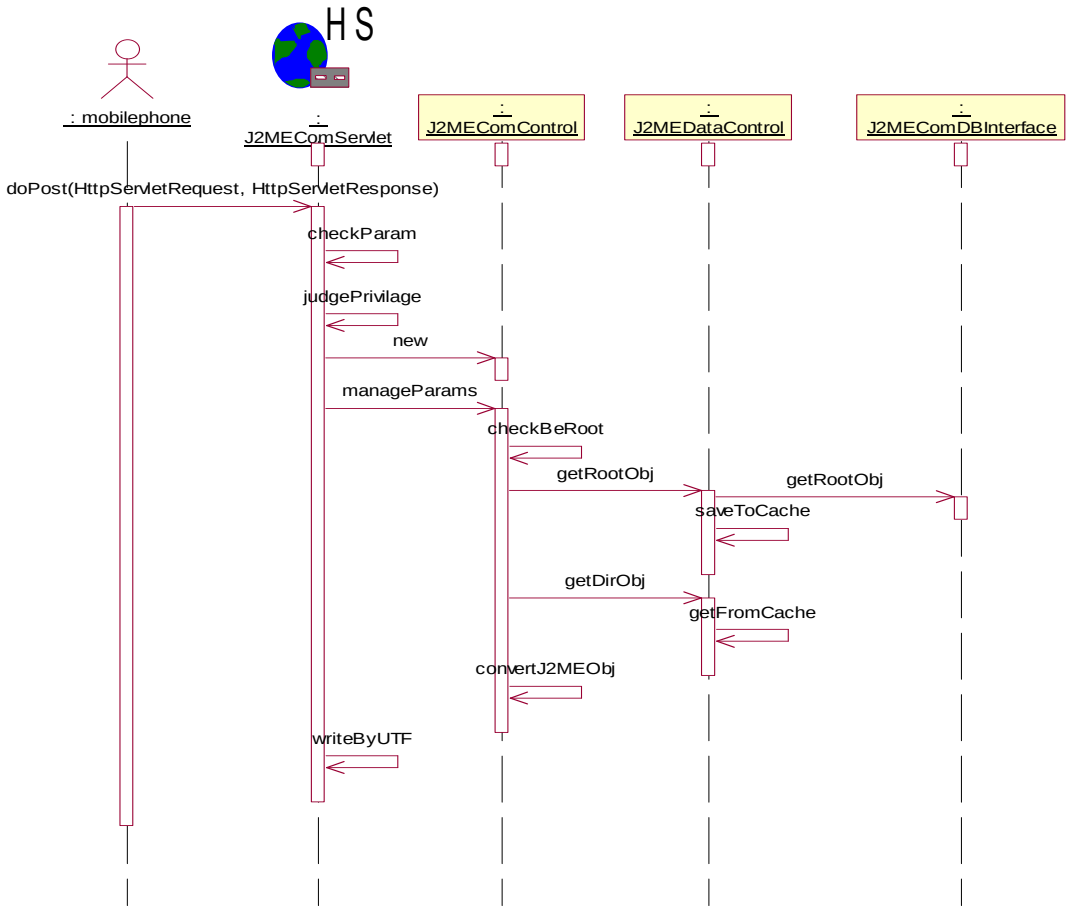


Figure 15 On-line client accesses to the root directory of online timing diagram

The client access to sub-menus, as well as books profile information, content, information flow is similar. As seen from the figure above, after the platform gets the root directory information, it is saved to the cache. The platform has a separate management thread for the cache. In accordance with the prescriptive time (half an hour) the cached content is updated. When accessing the book

content, the contents will not be saved to the cache inside because of the huge amount of data.

The following sequence diagram shows the procedure of accessing the books contents:

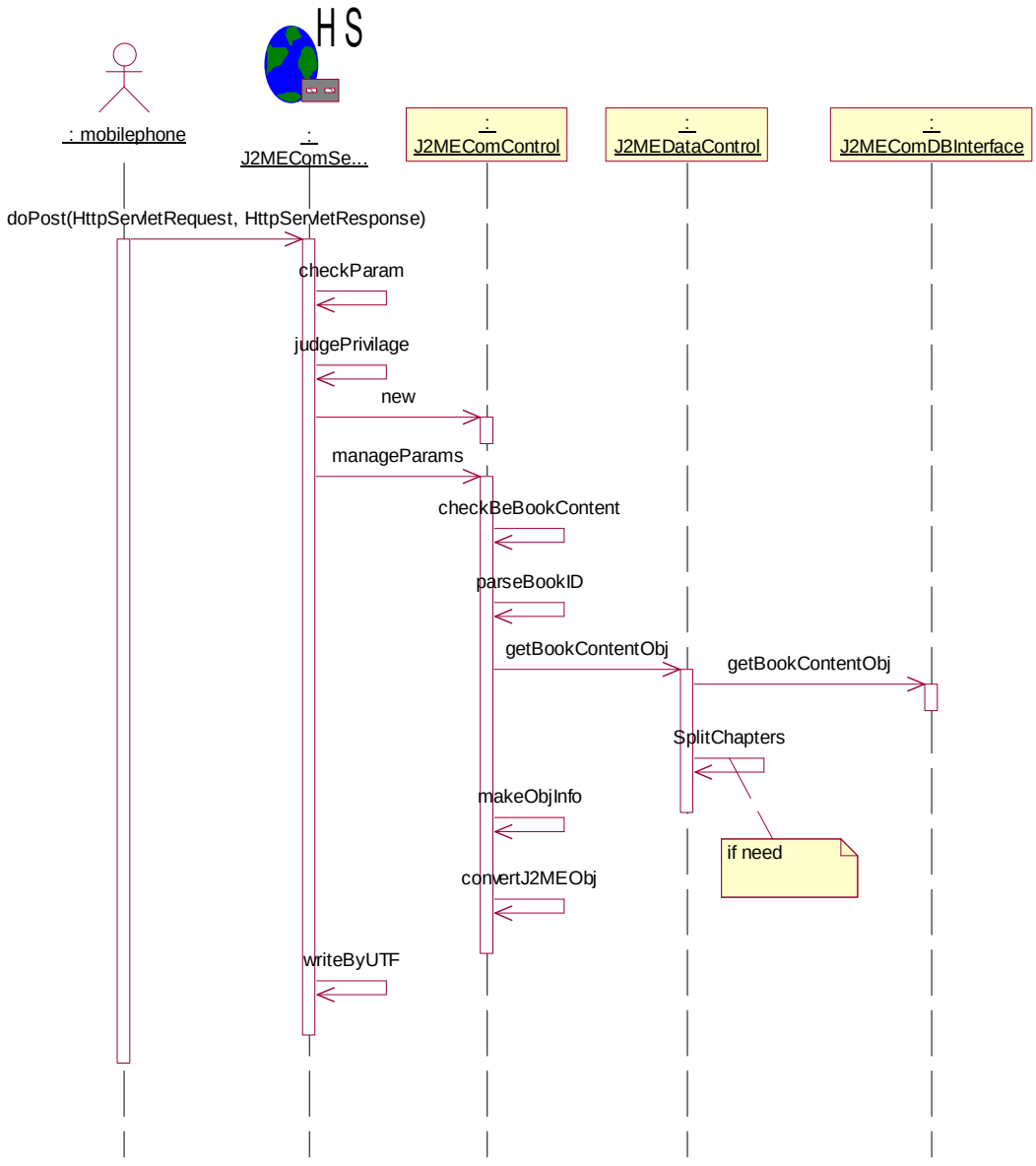


Figure 16 Timing diagram of obtaining books content

In on-line services, all the information the client can see is specified by the server, and the server can increase, delete, change the menu information, chart

information, books profile information, book recommendations information, book friends' reviews information, and book content. In this way if there are no major changes in the provision contents, it is not necessary to update the client, but just only the services.

3.3.3 Payment of sub-processes

The sub-system is responsible for parsing the customers' payment request and records, and sending the customer to a separate payment platform which connects to a bank network platform. Due to the payment platform security and confidentiality and stability requirements, the payment platform uses the SSL mechanism, and is independent of this system. We have described the payment subsystem here which does not include payment platform.

The following figure is the payment timing diagram:

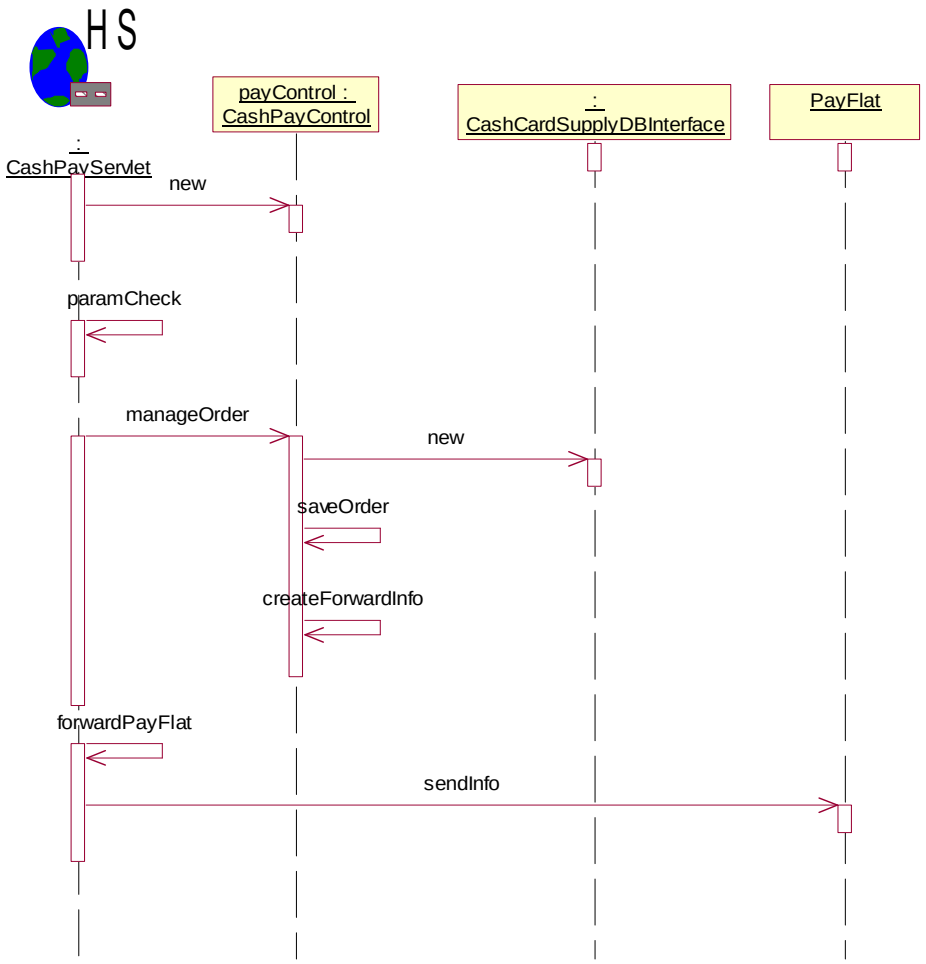


Figure 17 Payment timing diagram

First, the client sends a payment request, and then the interface layer initializes the control class. It passes the examined payment parameters to the control class, the control class orders processing, and then saves the information to the database, and builds the payment information of the payment platform. Finally the interfaces forwards to the payment platform.

After that, the payment platform processes the order, and guides the user to select banks and enter customer information. When the payment has been completed the sequence diagram is as follows:

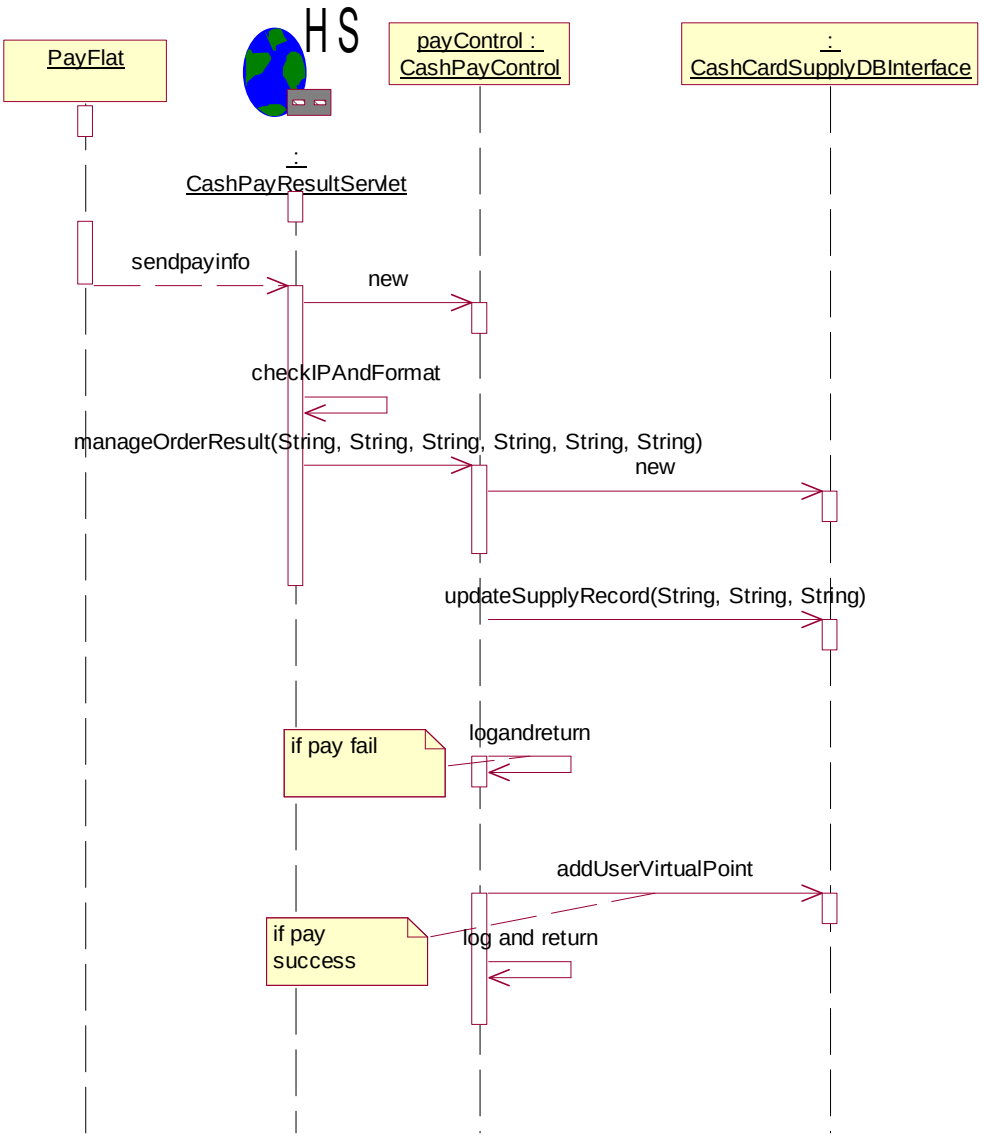


Figure 18 Timing Diagram of Order Processing

After the user finishes the payment, the payment platform will return the appropriate result information; the interface layer obtains the results information, and creates a control class. The information is passed to the control class; the control class analyzes the results. If the payment fails, it then updates the database, order records, and the user record. If the payment is successful, it updates the database record to be successful payment orders, and increases the relevant bonuses into the database.

Here the number of gold coins can be changed dynamically by customer service through the customer service system. The customer service system can check the real-time status of every order, and the changes in the number of gold coins.

3.4 Interface Design

3.4.1 Web interface design

The WBOOK Server Web interface adopts an easy-to-extend Struts framework for design. Struts is an MVC(Model View Controller) implementation. MVC weakened the coupling between the business logic interface and the data interface. It allows a more varied view layer. In Struts, the Controller is a Servlet, called ActionServlet. ActionServlet is a generic control component. The control component provides all the entry points of processing HTTP requests sent to Struts. It intercepts and distributes these requests to the corresponding action class (these actions are a subclass of Action). Otherwise, the control component is also responsible for filling the Action From with the corresponding request parameter (often called FromBean), and passing to Action Class (called ActionBean). Action Class is at the core business logic. It can access Java Bean, called EJB. Finally, Action Class passes the control power to the follow-up JSP file, which generates the view. All of these control logics use the Struts-config.xml file for configurations. The configuration file can add the front functional modules and modify the logic-orientation easily.

3.4.2 WAP Interface Design

The WAP interface dynamically generates wml page by using jsp. As there are some differences in the current phone supporting the wml mark, WAP uses the wml1.1 standards. So the WAP cell phones have a better way to browse the Internet. The WAP station directly connects to the terminal off-line automatically generated subsystem to complete the book production and downloads.

3.4.3 MMS Design

MMS services are mainly performed by the server-side. As long as the cell phones support MMS, the client/user can connect to accept our e-books. The MMS assembles the SMIL approach for adoption. The contents of books are obtained by the BookService system, and then assembled into a sub-system which sends a MMS. This step is done by a front action called MMSBookHelper class assembly MMSBookNode, and sent to the MMS sending subsystem to complete.

3.5 The system requires Runtime Environment

- The configuration of system requirements are: PIV2G microprocessor or higher, more than 2G of memory to run the server
- The server's operating system is Windows2000 (SP4), Window2003 (SP1), Unix, Linux
- The server database system uses SQL Server 2000

  The software which could be supported: development tools for Jbuilder2005;

  Version control tool is VSS 6.0 (SP5);

  Design tools: Rose 2003;

  Database design tool: Power Designer 9.5;

4. Design of the wireless e-book client

Compared with WAP, MMS, and Web mode, the terminal can provide the user with more convenient ways of browsing books. The terminal is also very easy for the client embedded inside the phone to be a value-added service for the manufacturer or operator. From a market point of view, the terminal reader can be used easily. It does not need to enter the address to browse all kinds of books on the download platform.

4.1 Total design

The off-line version is divided into two sub-systems: the Application sub-system, and the Read sub-system. The content is stored in text format. The off-line version uses stream to immediately access books content when reading. It does not load too much content into memory and is suitable for wireless terminal types of small memory devices.

The on-line version is divided into three sub-systems, Application, Read, and Download. Since the standard J2ME does not support the file creation and write operations, Persistence Storage only supports database format, it can not be used to keep books in text format.

Here is the system use case diagram:

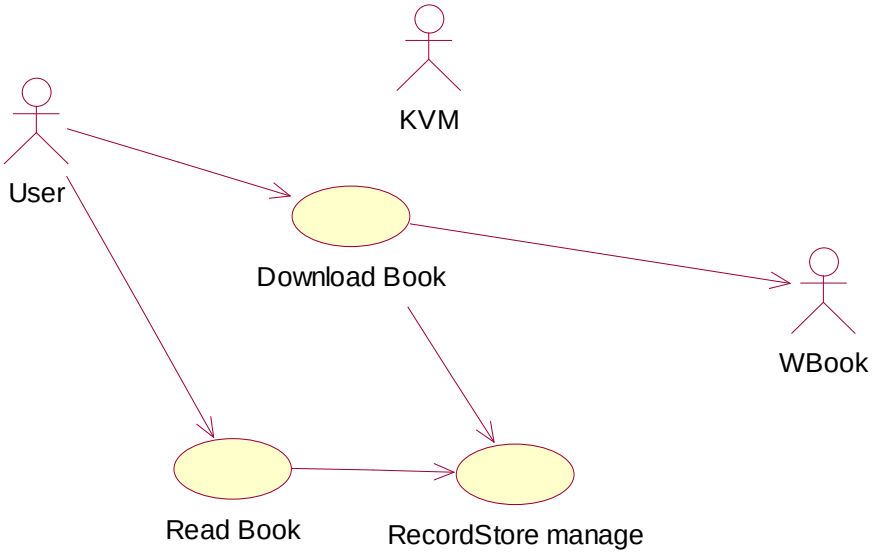


Figure 19 Case Diagram Used for offline reading

There are three use cases: download books, reading books, managing books. Three sub-systems are thus classified. The download sub-system is responsible for WBOOK platform to connect and access information. The read sub-system is responsible for providing user-friendly interface for reading, application logic and the record /store sub-system is responsible for the overall operation.

The sub-systems are illustrated as follows:

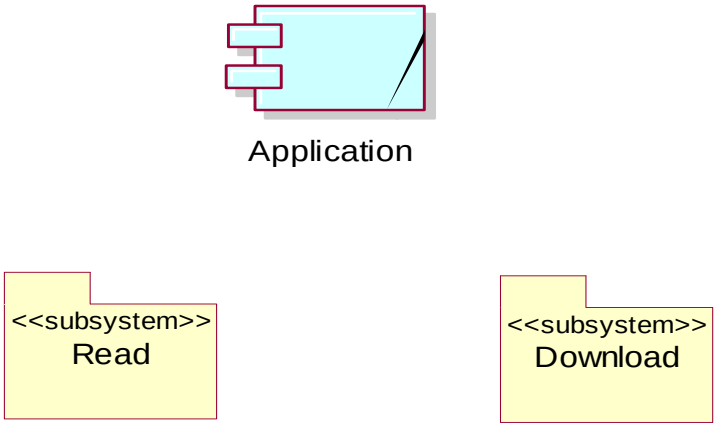


Figure 20 Sub-system diagram

The read sub-system is responsible for the content of books related to operations, including the user reading, bookmarks operation, the font settings, etc. The download sub-system is responsible for downloading books from the platform and in accordance with a definite pattern to the device which is stored in the database. The application subsystem is the system's main logic, including tools, midlet and so on.

4.2 Sub-system Design

The largest difference between the off-line version and the on-line version is that the on-line version has an on-line download module, and the way that the read sub-system obtains books content.

4.2.1 Off-line version

The operating objective of the off-line version is to produce text content in advance on the WBOOK platform. Hence its subsystems are simpler. The e-book client of off-line version is composed of two parts. One is the contents of books, and the other is an off-line reader. When the two parts request to download a server related to a book on the wireless terminal device, the WBOOK platform automatically generates Java programs according to the wireless terminal device type. The advantage of the off-line version is that after downloading once; the wireless terminal equipment does not need to re-connect to the server. Each read operation is run fast directly in the local run. The disadvantage is that when the book content changes, it is necessary to download the book content and reader again.

Here the class diagram shows the dependencies between places of various classes.

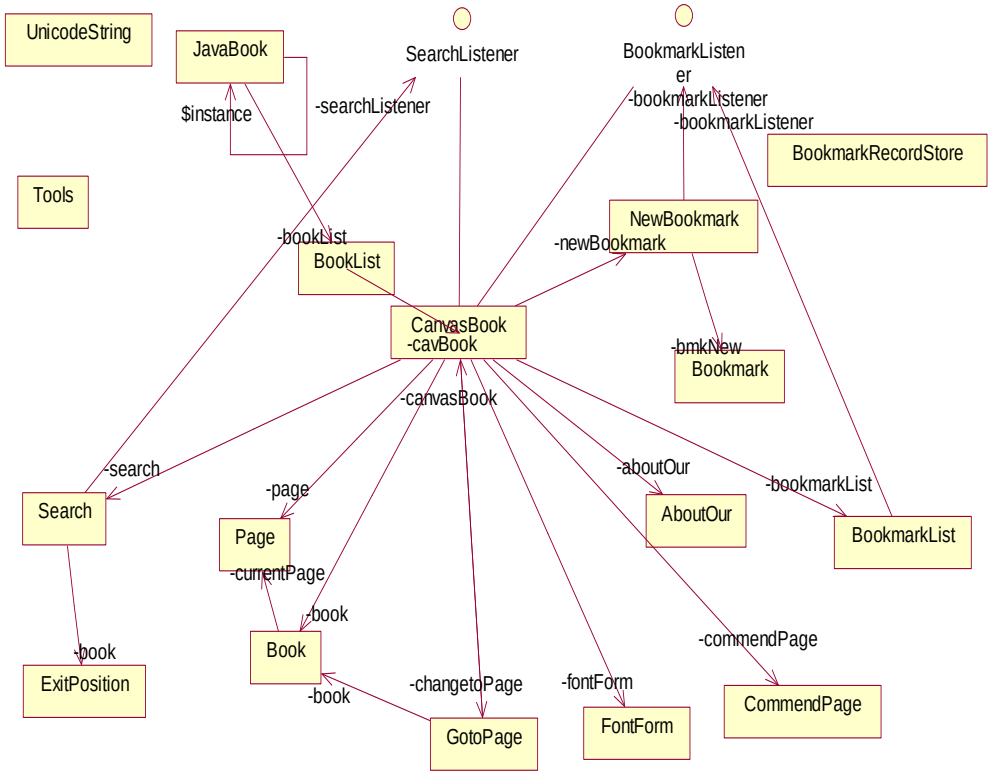


Figure 21 The dependencies between places of various classes

Here we can see that the BookList class and the BookPage class are used to control the books information. The interfaces are controlled by the CanvasBook class. In the off-line version, the books have been included in the package inside the jar. There are no classes to manipulate the database, but rather there is direct manipulation of text files. The off-line version provides a set of functions to read properties, such as: changing the screen background color, font size, color, auto-scroll feature, and bookmarking.

4.2.2 On-line version

The on-line version needs to exchange information with the WBOOK platform at any time. So its sub-systems are more complex. It achieves mainly a similar function of the Web-browsing books in the wireless terminal.

4.2.2.1 Application Sub-system

The application subsystem achieves the program's logic, and leads the user to use the system. When starting the process, firstly, a flash screen appears on the screen, splash screen images can press any key to skip, after the splash screen is the main menu, the user can choose to read books, or download the book, look at help on the information, or quit.

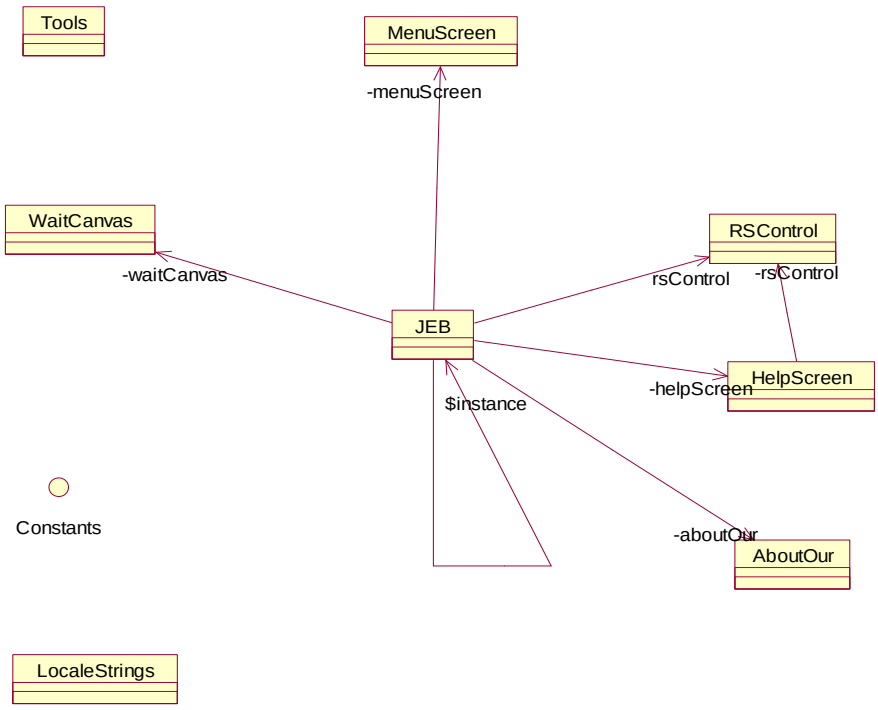


Figure 22 Application Sub-system diagram

JEB is the program entry point as well as the MIDLet class. J2ME hides the main method replacing the MIDLet class startApp as entry point. JEB data initializes in the constructor function, and begins to run the program in startApp. JEB has a static variable instance to store its own pointer. In this way, the other sub-systems could use KVM directly and easily. KVM is also due, and deals directly with some of the interfaces that need to use the MIDLet pointer, so here JEB provides the main logic of the program. MenuScreen is the main menu, and provides processing logic; RSControl provides access to the RS. It also provides Tools to encapsulate the method used in some systems, Constants packages

some constants, and LocaleStrings provides the local language. A help interface has also been placed here to download and the read interface is a separate interface for the two sub-systems.

Details of the main class diagram are shown as follows:

Entry categories:

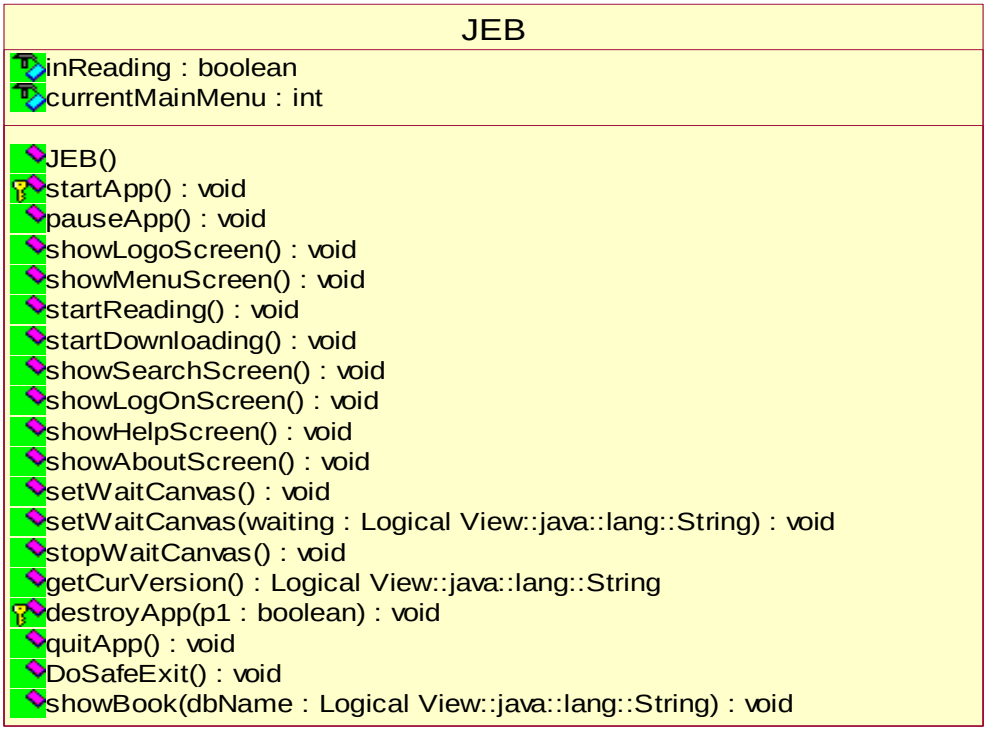


Figure 23 JEB core classes

DoSafeExit is the interface for safe exiting. This interface releases all occupy system resources and memory. This will not lead to the collapse of the hand-held devices.

StartApp (), pauseApp () and destroyApp () are called by KVM. startApp () is the entry point, which is the first entry in the program, but it is also called when the resumption by KVM occurs automatically when calls, SMS and so on have been interrupted. PauseApp () is the interface when a running program was

interrupted by KVM. Here it saves the program to run on-site, in the startApp () to determine whether it has been stored on-site for recovery. After the program has sent out the quit message, KVM calls destroyApp () to release system resources.

JEB has an inner class LogoScreen because the flash screen has nothing to do with the other; it only appears when the program starts, then it enters the main interface, and JEB appears as an internal class.

The flash screen is a separate thread. FILES [] are saved a series of pictures that the flash screen needs. When some key responds, it display the next flash screen picture, the last one shows the main interface.

The main menu class is MenuScreen. It adopts depiction to increase up to the Canvas control. At the same time, all the menu buttons perform the corresponding program logic.

The figure below shows the RS control class:

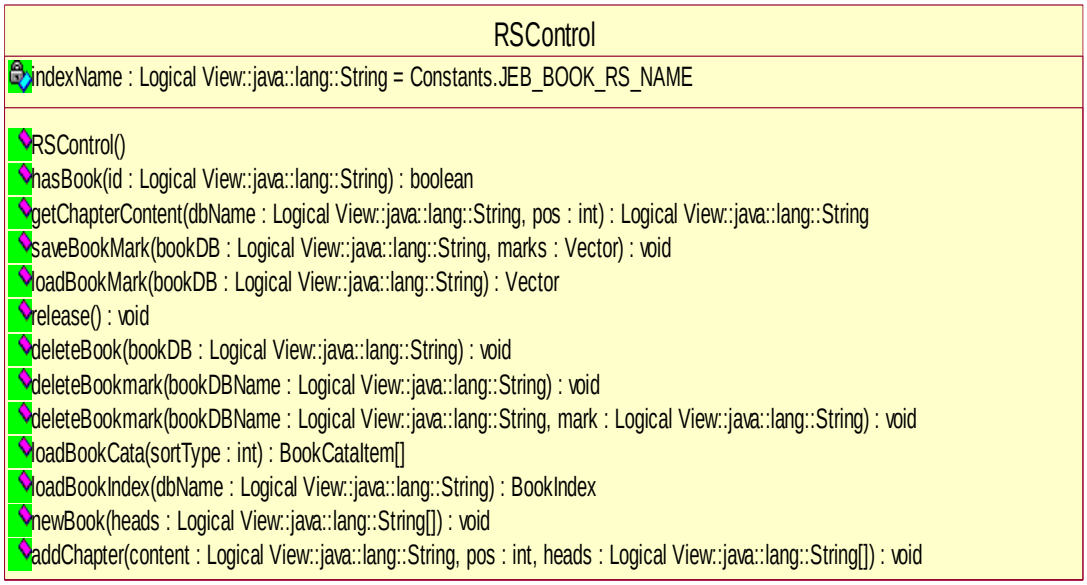


Figure 24 RS control class

In order to operate the hand-held device and have the database facilitated, it is necessary to establish the RSControl class to encapsulate their respective operations and store and read data in accordance with the defined data formats, so that outsiders do not have to know the data format. As the hand-held device database is only a simple database table, here the operations are operated for byte stream. The byte stream facilitates the definition of a variety of data formats, so it is necessary that the entity classes have achieved the Serialize interface.

Help interface class: HelpScreen. The help interface provides a variety of solutions to the problem, so there is an internal class HelpInfo. It has only two properties, quest, and answer, while the HelpScreen lists a variety of questions. And, when clicked, it displays answers to questions. The problems here are built in the system, and can be updated on-line, when the user has new questions, it is possible to connect to the requested WBOOK, and then update the issue.

Tools Class: Tools packages many unrelated logic operations. The greatest goal of the Tools class is to re-use, so the unrelated logic operations functions will be added to the tool class.

The figure below shows the Constant interface class:

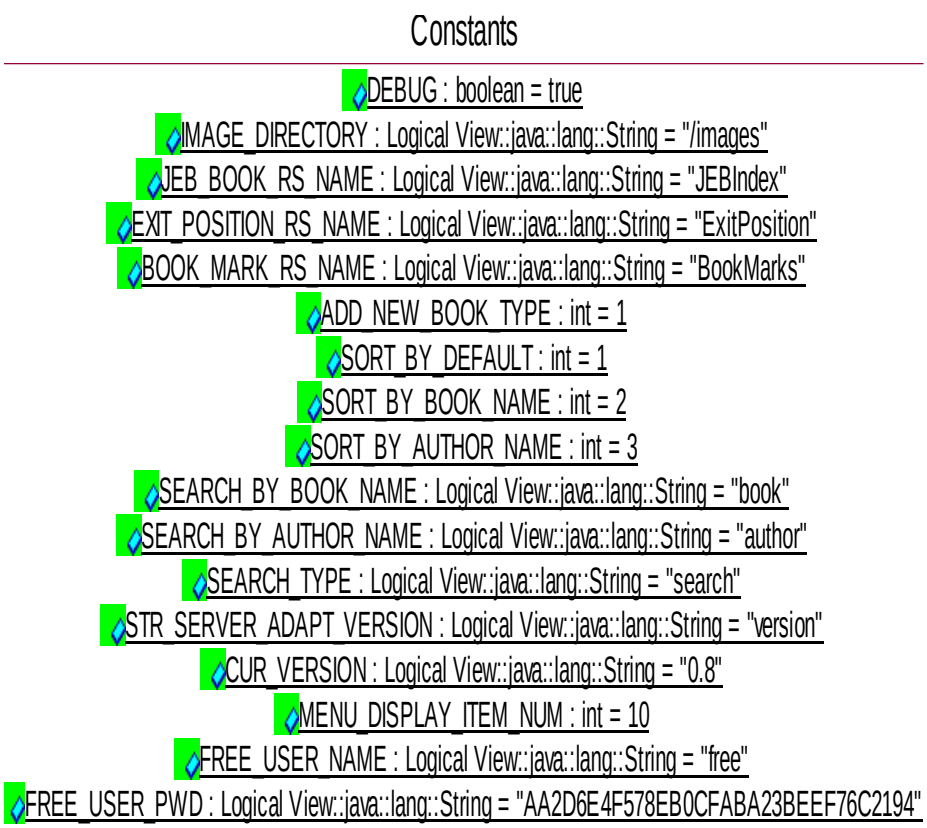


Figure 25 Constant Interface classes

This is another usage of Interface. By defining a number of static constants we can achieve the overall use. The DEBUG parameter can determine whether print debug logs in the system. FREE_USER_NAME and FREE_USRE_PWD are the system built-in free of charge user's user name and password, where password is encrypted by MD5, but the official users will ignore these two parameters.

Localization Language classes:

Figure 26 Localized Language classes

In order to modify the system language conveniently, here we establish a LocaleStrings class along Windows System Character Map. All the localized text is placed inside, so when the user changes the language version, only one class needs to be changed.

To wait for the screen type, we use WaitCanvas and Animate:

Animate is the WaitCanvas inner class.

It needs to emerge??? animation in the waiting interface. Animation control is achieved by internal class Animate, and the animation speed is specified in the external class. The animation here is just a painted progress bar.　Waiting for

screen requires a new screen called WaitCanvas, the stop interface to stop the animation thread work. Otherwise, there will be more than one threads running, multi-consuming system resources.

4.2.2.2 Download Sub-system

The download sub-system leads users to download content from the WBOOK. After entering, it shows “download the root directory first”. Downloading the root directory includes new books download, classification download, list information, etc. After the users select the appropriate content, they could access to books lists or sub-menu, and then the users select their book of choice and the book’s profile information is displayed. Users can also choose whether to check the recommended information, the book friends’ comments, and then confirm whether to download. If the phone has downloaded the book, it will be asked whether it would like to download the cover off of the original book. The download speed may be affected by the speed of the local network.

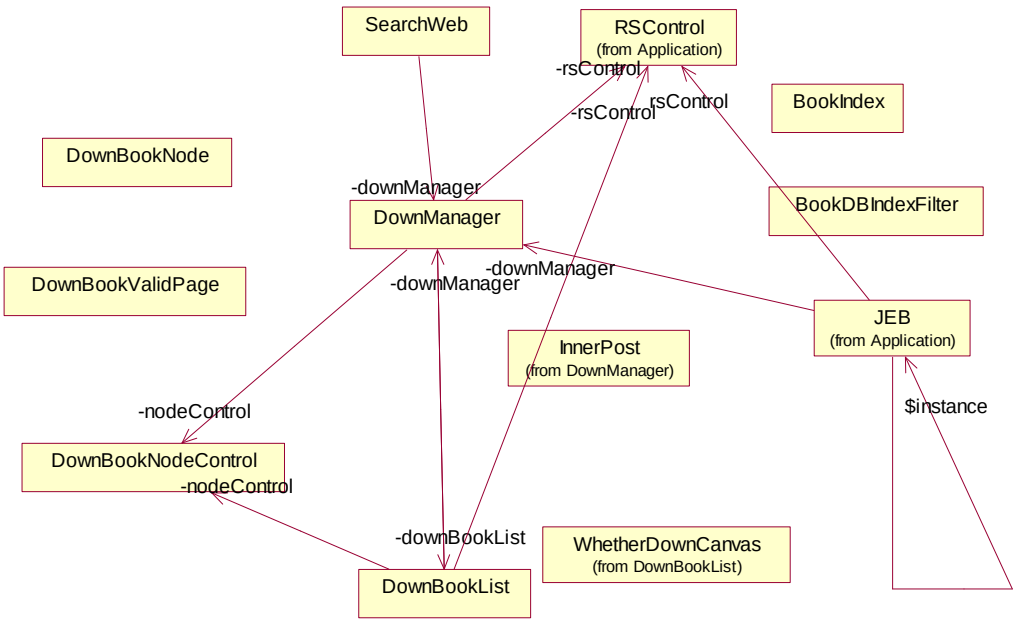


Figure 27 Download Sub-system

The download sub-system still needs to use RSControl to save the downloaded data information to the database, but the menu information, as it is a small amount of data,is stored in the cache to speed up the run rate. The connection is completed by the DownManager internal class InnerPost.

Details of the connection are described as follows:

InnerPost is responsible for sending post and it is the internal class of DownManager.

DownManager is responsible for the connection, and provides different external connection interfaces.

In the J2ME platform, the connection process must be achieved with the sub-thread, or if network is experiencing problems, it can block the main thread, causing crashes, where the sub-thread appears as an internal class, that is InnerPost:

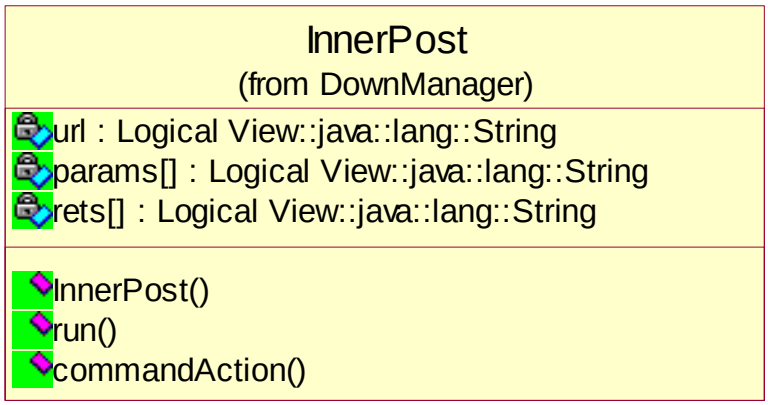


Figure 28 InnerPost class

After processing the received data, it also works in run in InnerPost. The processing result is used to display and apply by DownBookList.

DownBookList displays the contents of the menu items and menu items through the List and the Form. As for whether or not to download some menu items, it has also a built-in confirmation screen, which is WhetherDownCanvas.

DownBookNodeControl achieves the download of data control and buffering mechanisms are also included in this control inside the class.

Downloaded entity class: DownBookNode. The network allows s the UTF byte stream to re-structure for such an instance in order to facilitate the application.

Similarly, in order to confirm whether the downloaded books contents of interface as a separate class exists, here after the confirmation call, DownManager downloads the interface again.

After downloading, the data needs to be saved to the device database, then we can use the data interface layer of the control class RSControl, as it offers convenience of storing the data in the byte stream. The use is a class structure, and the class structure is the entity class, and provides an interface to serialize and de-serialize the interface.

The class diagram of these entity classes are as follows:

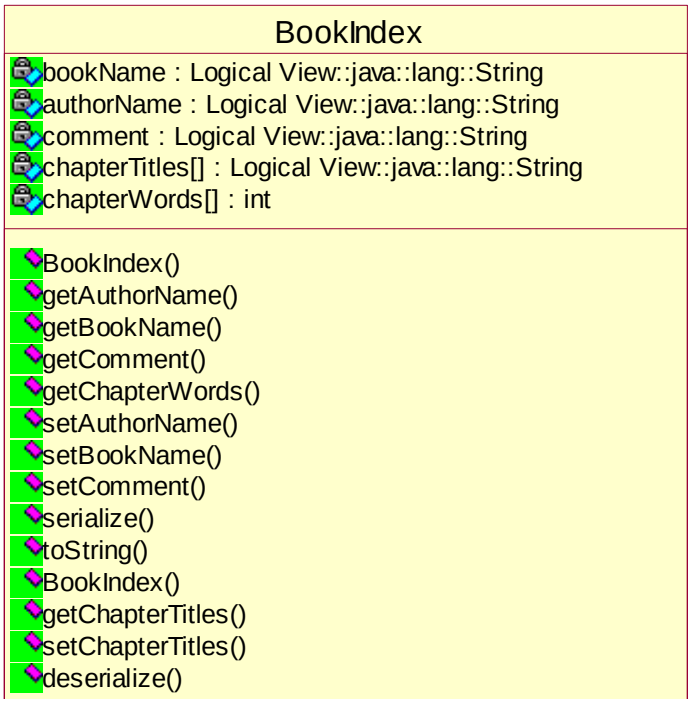


Figure 29 BookIndex class

BookDBIndexFilter is the filter class for BookIndex index. It should select the records through the filter class to in accordance with a format to meet the requirements in the numerous database records (byte stream).

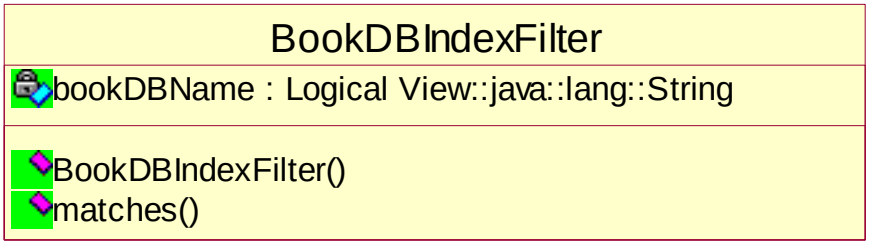


Figure 30 BookDBIndexFilter class

The matches () class is called by JVM Java Virtual Machine during the database query according to the return value to determine whether it corresponds with the filter conditions. Here the filter conditions are the specified book databases names are the same.

To make it easier for users to find the books they are interested in, class SearchWeb provides the search interface.

Search requirements of their on-line connections, and the search results are displayed to the user, while the connection and display operations are controlled by the Down Manager.

4.2.2.3 Read Sub-system

Read sub-systems:

The most important sub-system is the reading subsystem. It displays the books directories that have been downloaded on the cell phone first, and provides the management features of books (such as delete, sort), if there is no book, the user is prompted to download. For each book, it displays the profile information first, and then it reads it.

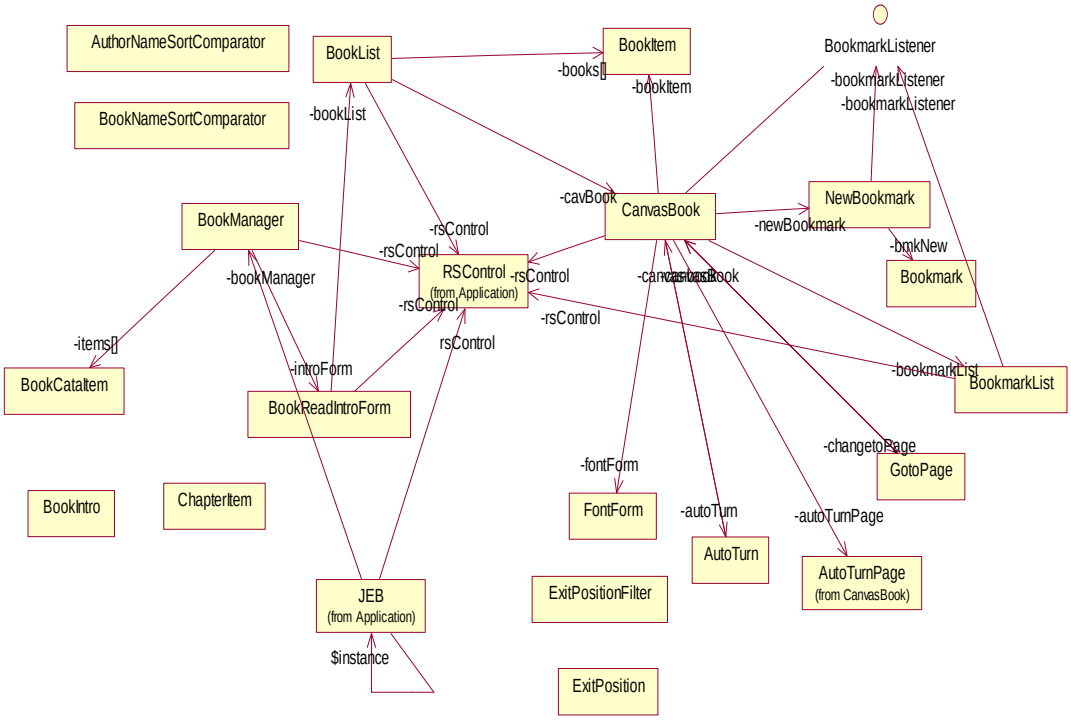


Figure 31 Read Sub-system

The cores of this sub-system are BookManager and CanvasBook. The former is responsible for the display of the list of books; CanvasBook is responsible for the content of the display of books. They all use the RSControl to read the contents inside the device database.

Book Manager is used to display the books list, and manage the list. Showing the books list needs is based on certain rules, according to book titles to sort BookNameSortComparator, or author name of the sort AuthorNameSortComparator, or download the order of ranking.

The sorting algorithm is bubble sort, and it is achieved by the “compare” interface.

In the books list, after the users select the books and then the cell phone can show the introduction of the displayed books by using the BookReadIntroForm function.

After viewing of the briefing of book, including author, words, relevant advertising, then the users could browse a directory of books; it is achieved by the BookList.

Here, the chapters’ titles will be displayed, and the user can select a chapter in a book and begin to read the books, but books are displayed in the CanvasBook class which is responsible for reading these works.

There are two ways of scrolling text options, one is operating direction keys, or the number keys to select a line or a page turned. The other is to select the auto next page, e automatically by default every five seconds per page. Users can set the page interval on page automatic settings. The auto next page is controlled within the class AutoTurPage.

AutoTurnPage also controls the time interval through a sub-thread. The auto next page is set by AutoTurn.

The reader also provides a way to modify the font from FontForm. The Font Settings page sets the font size, foreground color, background color, and style.

The reader also allows jumping to another page, using the the GotoPage class to achieve this function.

In order to facilitate the user to continue reading, the reader of this book has recorded the last reading position. The following figures show what the ExitPositionFilter class and ExitPosition class can achieve.

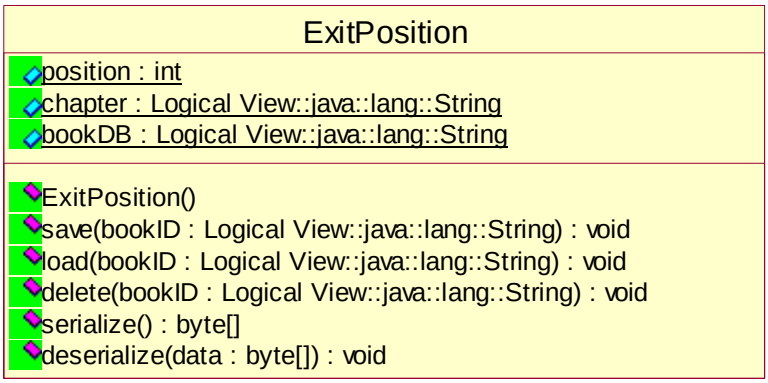


Figure 32 ExitPosition class

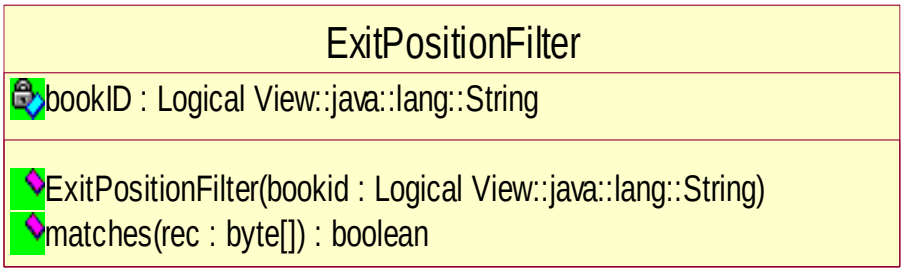


Figure 33 ExitPositionFilter class

ExitPositionFilter here is a filter class. It is used to look out the specified exit location in the database. It implements the matches interface.

As a reader, bookmarks are essential. The reader has prepared a list of bookmarks for each reader.

The Class diagram in this series of classes is as follows:

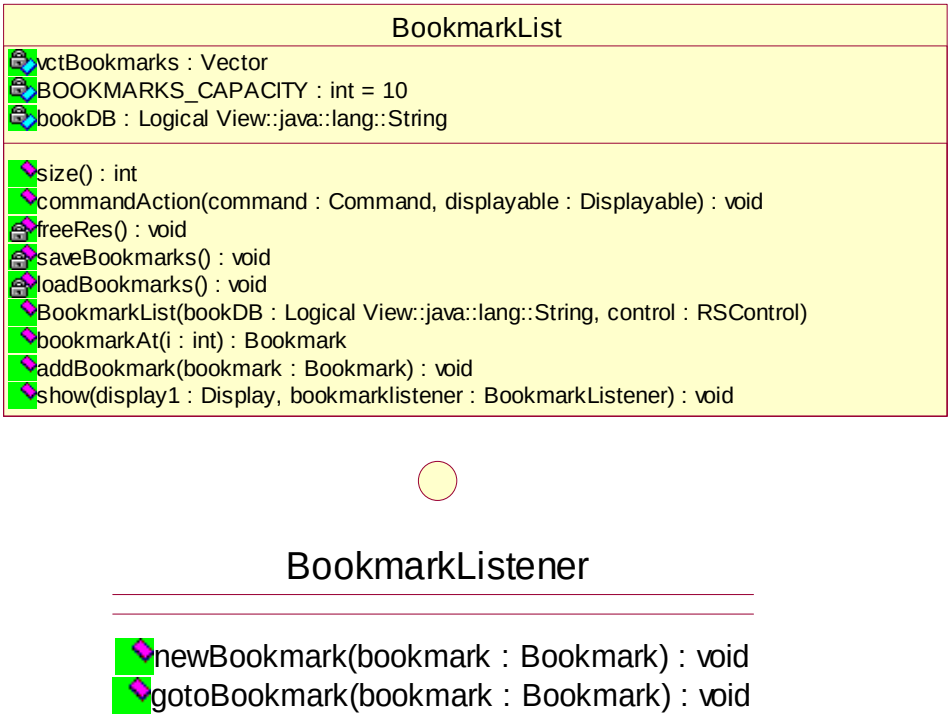


Figure 34 Bookmarks class

The following are several class diagrams of entities used to read sub-classes:

The BookItem class records the book file name and the file size.

The ChapterItem class records sections information

The BookIntro class records the book title, author, chapter, quantity, etc.

The BookCataItem class records the Book Classification Project.

4.3 Data Definition

4.3.1 Internal Data Definition

J2ME persistent storage does not support file formats. It can only be stored in the Record/Store which is a simple database and it does not support multi-line multi-column. It is equivalent to only one column of the table. Therefore, we need to establish a RecordStore database to store the names of books, and each book is to create a separate RecordStore to respond.

Here is a Database of RecordStore:

Database Name: JEBIndex
Usage: Storage of books name, one book on each line
Format: Title + author's name + book + books database name + remarks
Books database name: in the above table are defined
Usage: Storing all the contents of this book

Format:
The first line: the content index within;
Including: Title + author's name + Notes

The second line is the contents of book chapters, each row has a limited number of words within the allowable range (tentative 40K), if a chapter does not fit, it is split into multiple chapters, and consecutive numbers are added as marks at the end.

Format: chapter name + content + Remarks
Exit Position Format
Database Name: Exit Position;
Each line is a record;
bookDB + chapter + position

bookDB is the name of the database of books, chapter is the chapter name, and position is the row number in this chapter.

Bookmarks:

Database Name: BookMarks

Each line is a record

bookDB + num + file + title + position

BookDB is the name of books database;

Num is the number of bookmarks for this book;

File specifies the bookmark of the chapter;

Title is the bookmark name;

Position is which line of this chapter.

4.3.2 Interface Data Definition

The users could click to download, even into the server URL

Getmobileinfo

Function: The client connects to this interface, through the parameters to download the content;

Parameter format: the first time readUTF reads the number of parameters, if it is 0, then there is no parameter, then it returns to the first level sub-menu.

The second time readUTF reads the contents of the first parameter; the third time is the second parameter of the content, and so on.

Return Data:

Number of parameters (BookNode type)

BookNode.getName + BookNode.getId

Loop

The ID, is the directory id +"."+ book id. If it is only directories, it is not the books, but only the directory id

4.4 Process Design procedures

This Section only deals with the online version sequence diagram.

We use the sequence diagram to describe the process flow. A complex process is described by activity diagrams and collaboration diagrams.

Now, the diagram of the system boot sequence will be explained:

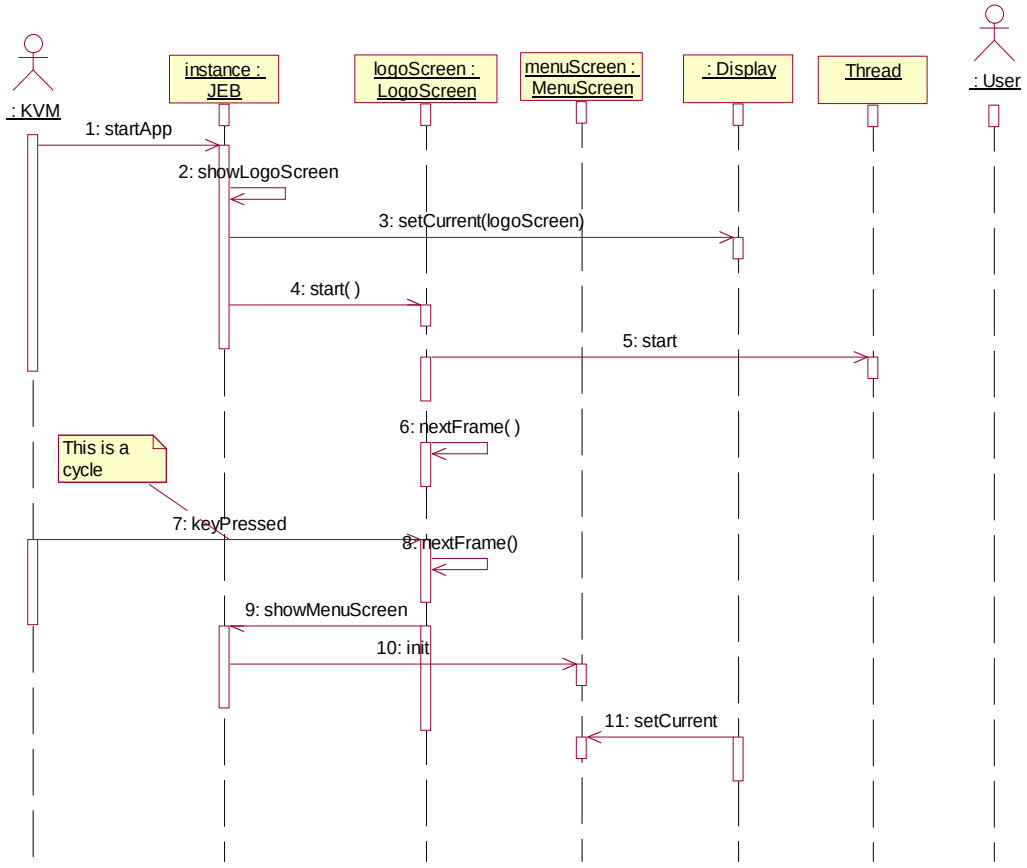


Figure 35 System startup timing diagram

It is the diagram of the system exiting sequence:

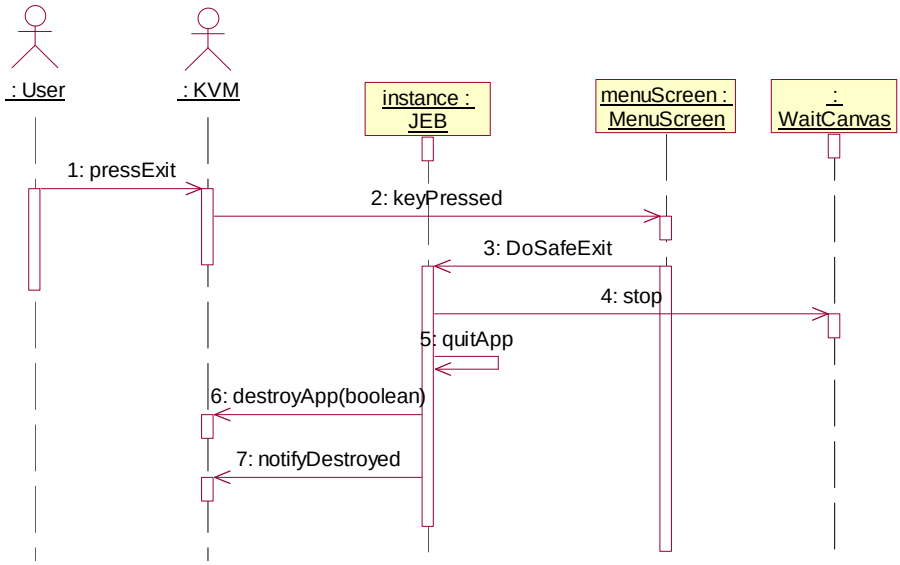


Figure 36 Systems exiting timing diagram

The download screen sequence diagram is shown below:

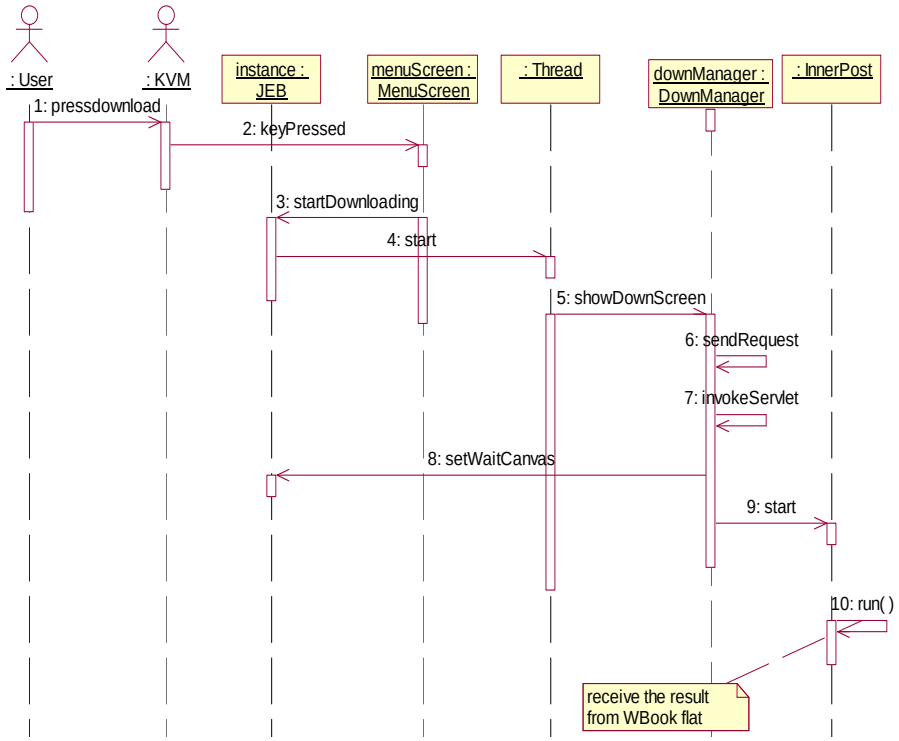


Figure 37 Download timing diagram

It shows the read interface Sequence diagram:

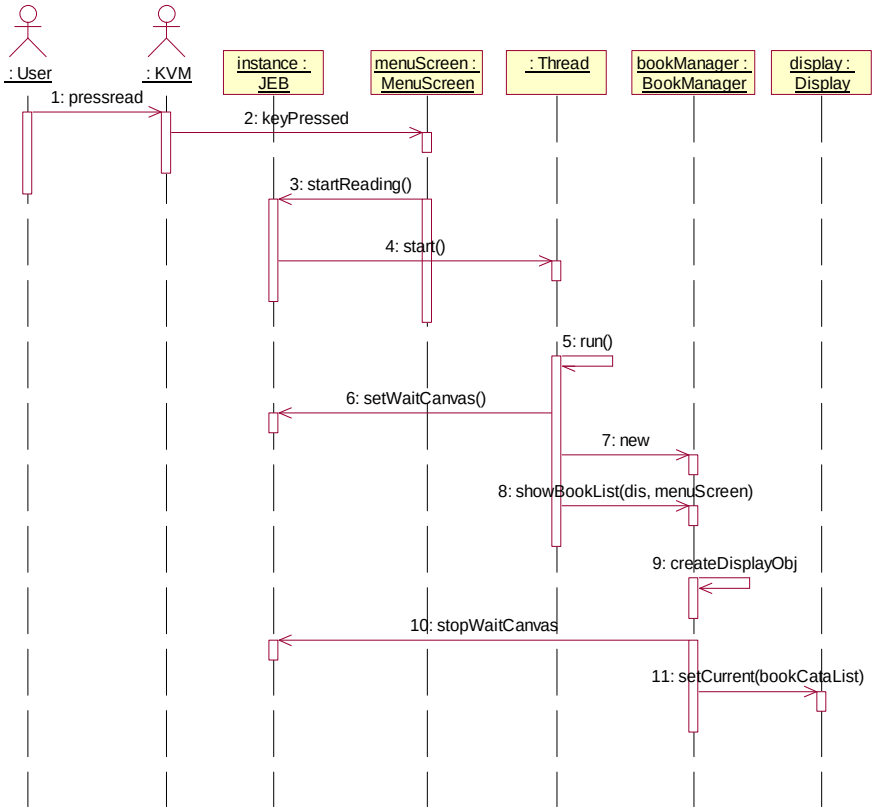


Figure 38 Read timing diagram

Figure 39 i shows sending data sequence diagram:

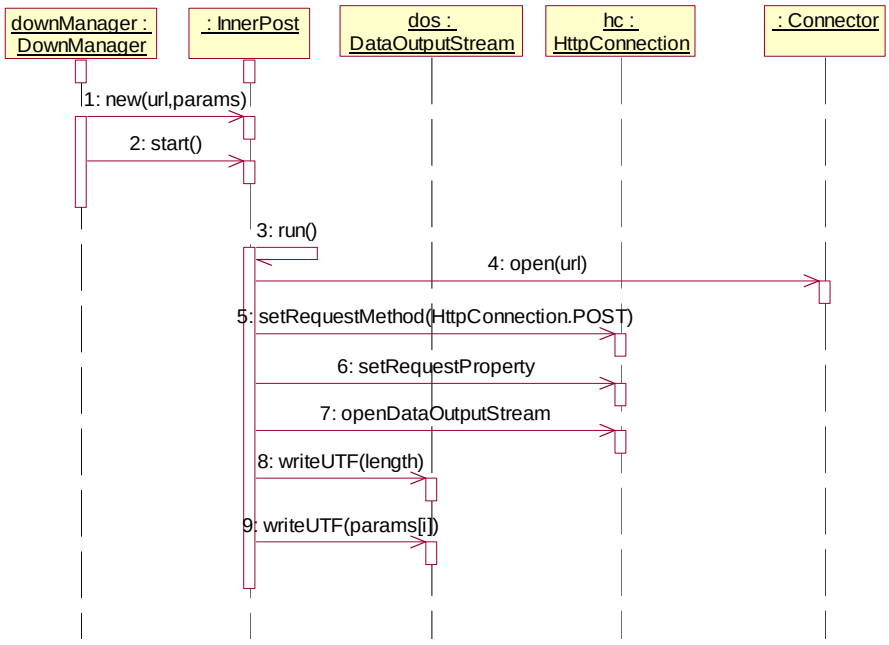


Figure 39 Data sequence diagram

Figure 40 illustratesthe receive data sequence:

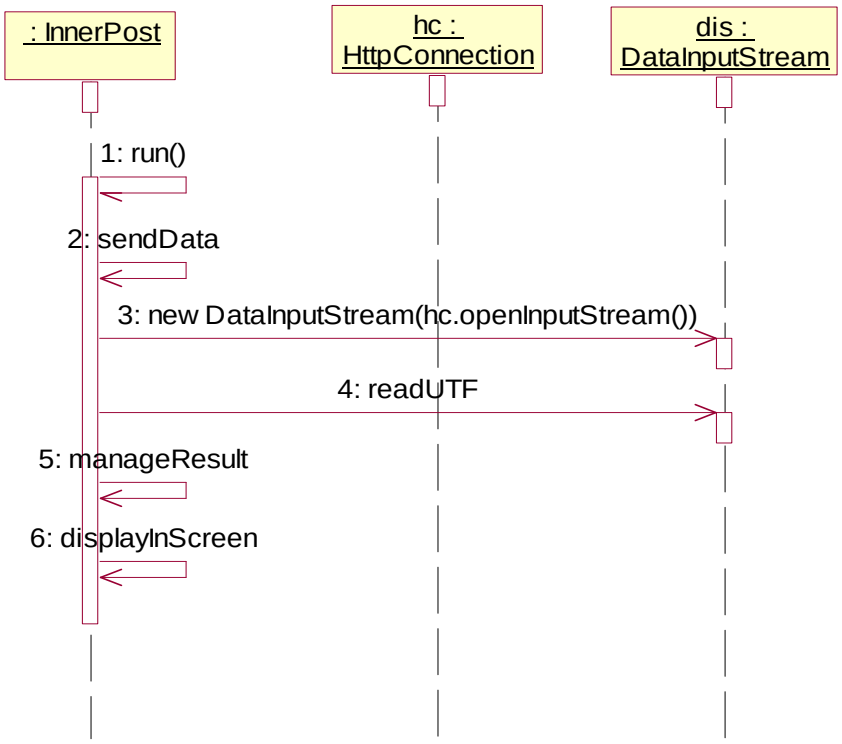


Figure 40 Receiving data timing diagram

A detailed flow chart is as follows:

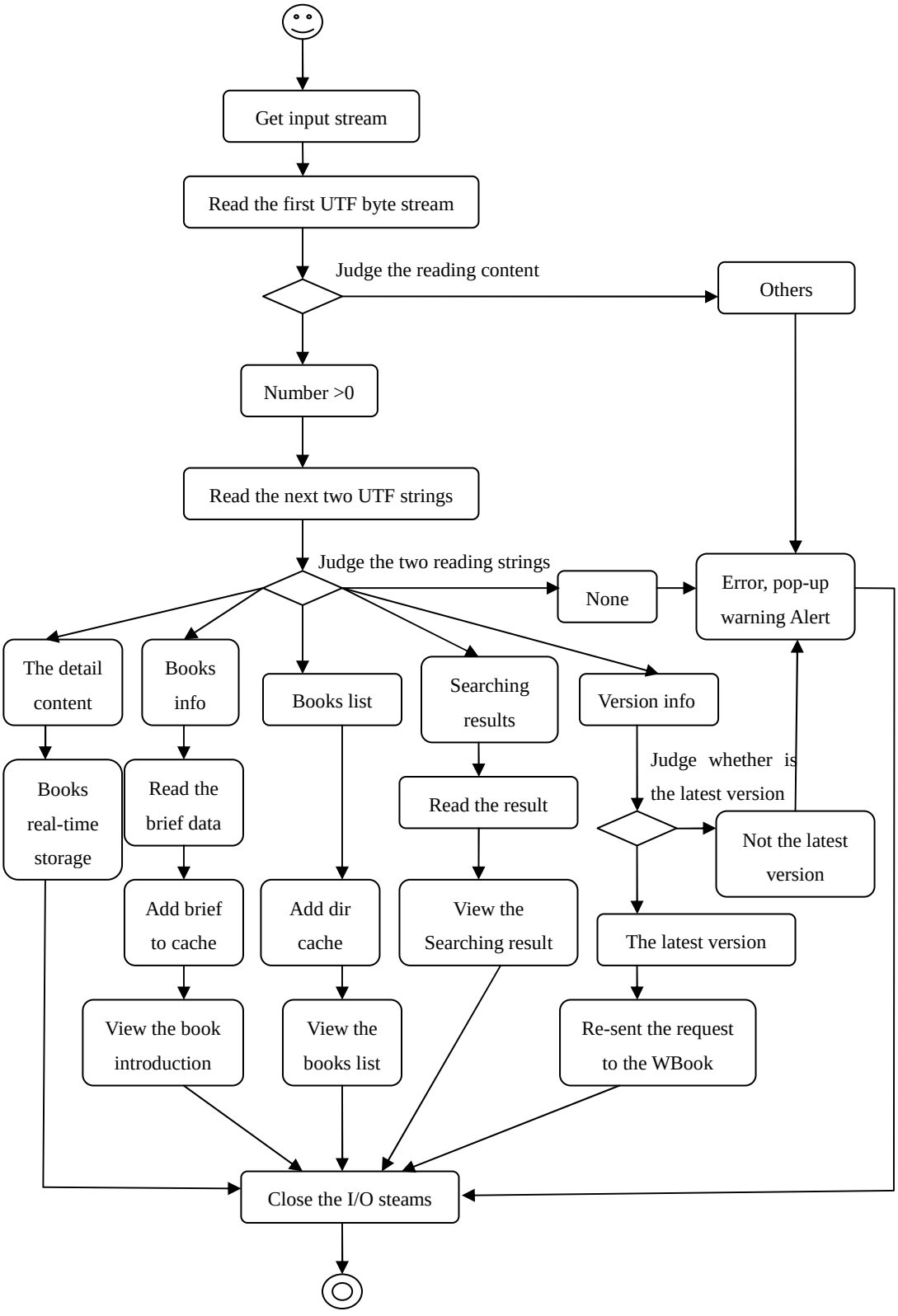


Figure 41 Detailed flow chart

The following flow chart shows the books content details stored at real-time as follows:

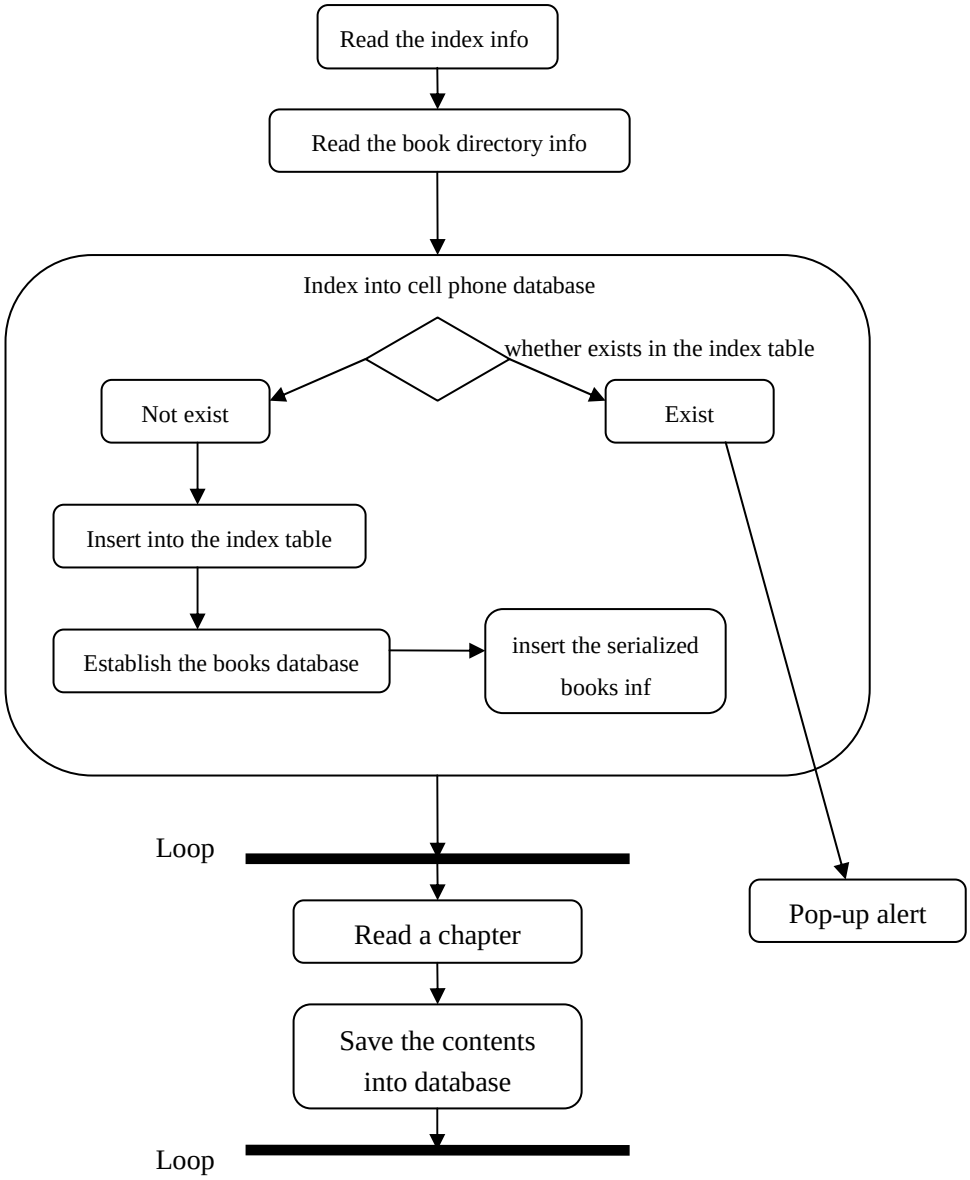


Figure 42 Books content stored at real-time

All above device databases use a serialized byte stream while storing.

The location of saving the last reading:

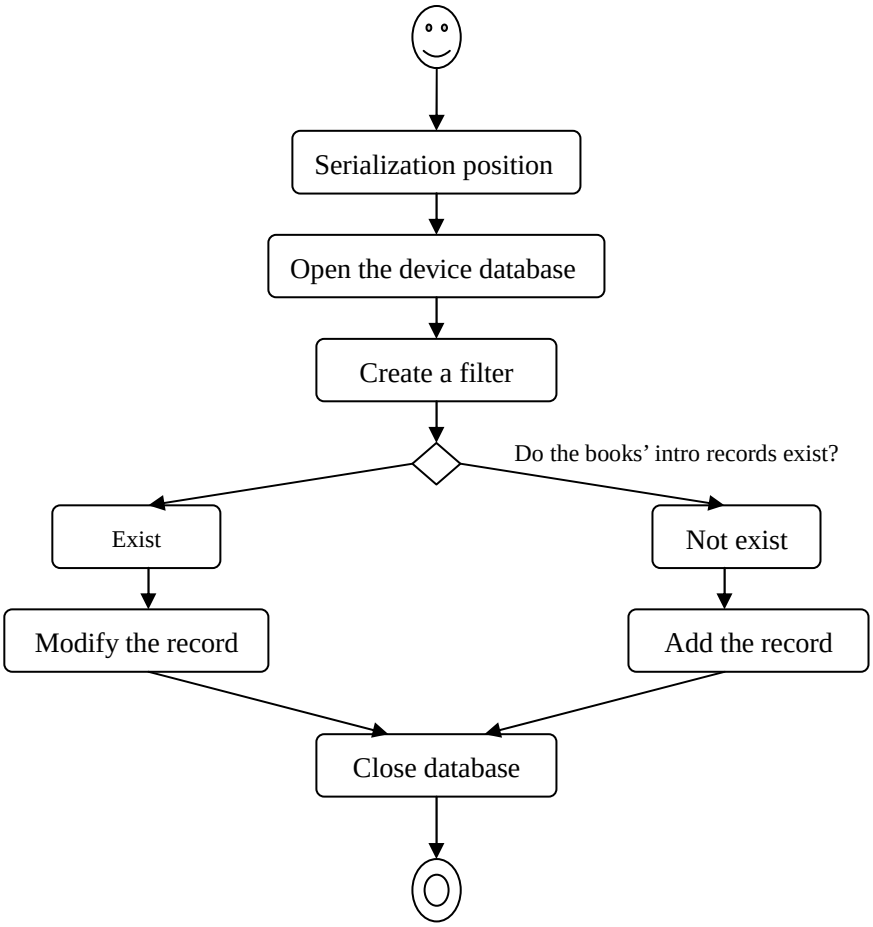


Figure 43 Location of the reading flow chart

The key of reading the exiting location is the federalization of the data in the equipment database.

5. Test Platform

5.1 Single test

The purpose of testing is to discover the platform shortcomings. It is a process for finding out mistakes to implement procedures. In order to build a scalable wireless book value-added platform, a platform is integrated by many small modules. So we need to make sure that each module, each class can work normally. It is extremely necessary to carry out the monomer test. The monomer test here is mainly fine-grained tests. In order to improve development efficiency and achieve automated unit testing, we mainly use JUnit for testing.

JUnit is an open source code Java testing framework. It is very simple, provides immediate feedback, and is free of charge. JUnit is used for extreme Programming, which helps reconstruction, and there are assertions. We adopt such basic processes as: the idea -> write test case -> compile code -> test. So that when we find problems, we can quickly trace the cause of the problem to reduce the difficulty of the error correction regression testing.

The JUnit framework package includes all the classes that JUnit tests require. TestCase class is the core part of this package. We inherited this class to establish our own automated test unit. The other classes support the TestCase class. Their role is to collect the result, and aggregate numerous test classes to be a test suite, and provide assertion. We use another key Test to define the TestCase, and a common contract with Test Suites to provide a hook at the same time.

5.2 Stress Test

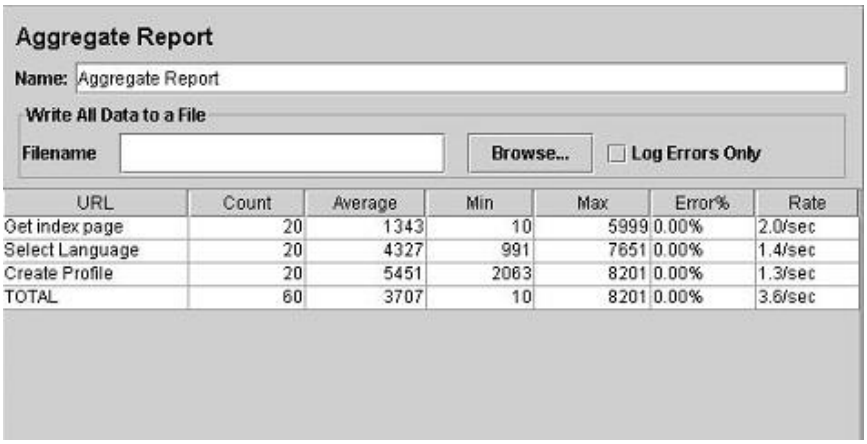
A platform may have thousands of people simultaneously accessing the case, so it is necessary to conduct stress tests.

Stress tests need to simulate thousands of people simultaneously accessing the platform, and to test if the platform functions under stress speedily and accurately.

Here we use JMeter for testing. JMeter is an open source Apache Software Foundation product, so it is completely free.

The following are examples of our test cases:

This is the aggregate figure.



URL	Count	Average	Min	Max	Error%	Rate
Get index page	20	1343	10	5999	0.00%	2.0/sec
Select Language	20	4327	991	7651	0.00%	1.4/sec
Create Profile	20	5451	2063	8201	0.00%	1.3/sec
TOTAL	60	3707	10	8201	0.00%	3.6/sec

Figure 44 Test polymerization map

The following steps are to determine the corresponding time standards:

- Users will not notice a delay of less than 0.1 seconds.
- Less than 1 second delay will not interrupt the user's normal thinking, but some delays will be noticed by users.
- If a delay is less than 10 seconds, the user will continue to wait for a response
- If a delay is greater than 10 seconds; the user will give up and start other operations.

The users could find the value through looking-up the table below:

confidence interval （probability）	Value Z
0.800	±1.28155
0.900	±1.64485
0.950	±1.95996
0.990	±2.57583
0.995	±2.80703
0.999	±3.29053

Figure 45 The results of data tables

By the values of the table, and then calculate 95% confidence interval [3.707-1.95996 * 3.6 /, 3.707-1.95996 * 3.6 /] to [2.796, 4.618]. The value basically meets the requirements in simulating the real environment in the laboratory to detect this value. We could obtain better results than the results in the simulated environment if the system was in actual use.

6. Summary and Prospects

6.1 Summary

With the support of Jilin University, our team could complete the development of the whole system.

The overall objective of this study is to establish a platform which can automatically match a multi-terminal with the mobile e-book platform. Users can use four ways to obtain the contents of books, namely WAP, mobile terminal readers directly downloading applications including books content, and then uploading the application to cell phones via the IR / Bluetooth transfer mode as well as MMS. Indeed, the platform should include a billing module, third-party content access and billing modules, subscribe / unsubscribe modules, an on-line payment module, and a GPRS MMS book on-demand / subscription module.

The current platform can adapt terminals automatically for more than 1000 cell phone models on the market and generate executable program correspondingly.

This thesis demonstrates the J2ME platform for wireless mobile library data value-added system obtains the relevant content that is based on the. The figure below demonstrates the interaction relationship between each module in the system:

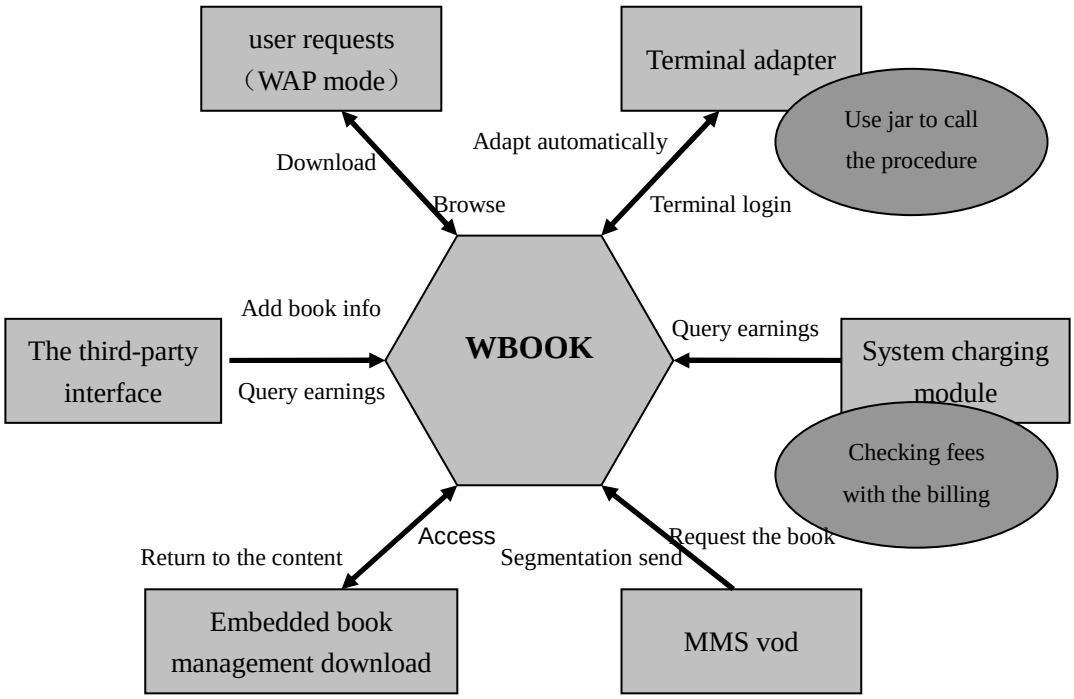


Figure 46 WBOOK platform interacting with the various system modules

6.2 Future Work

The project is going to be continued and the team is going to implement the following:

- Extend the CDAM Unicom network and China Netcom's PHS network.
- Use the data mining technology – get prepare to put in the exact advertisements

Each purchase will be recorded on the users use on the platform. According to the type of books the user downloads type over time, the user's buying preferences and interests will be determined. It is possible to analyze the areas where the user accesses the Web interface, through the cell phone number databases, and IP address records. It is also possible to classify different parts of business information, and push advertisements to the e-books folder page, or on the web page seam.

- Data “flows” between different devices

Since J2ME has cross-platform features, we are ready to test data between devices in a variety of "floating" possibilities. For example: we can buy an electronic book from the wireless terminal, and transit through our platform, that allows users read continuously on their own PC, car DVD / electronic navigation screen, PDA or smart home appliances. WBOOK is truly: anytime, anywhere, enjoying reading, enjoying life.

References

[1] Liao Jianxin. Mobile add-valued service developmental trend, Beijing University of Posts and Telecommunications, ISSN: 1008-5599, 2001

[2] Zhang Wei. J2MEwireless communication technology application development, Beijing Hope Electronic Press, 2002

[3] Chakravorty Rajiv, Katti Sachin, Crowcroft Jon, et al. Flow aggregation for enhanced TCP over wide-area wireless, Proc IEEE INFOCOM 2003, Piscataway, NJ: IEEE Press, 2003.1754-1764

[4] China Mobile Communication Corp. China Mobile M-Dream java technical manual, Referred on: 01-10-2004 Available at: http://www.monternet.com

[5] China Mobile Communication Corp. China Mobile MMS interface technical file, 01-10-2004, http://www.monternet.com

[6] Roger Riggs et al. J2ME wireless equipment Programming, Mechanical Industry Press, 2002

[7] Bruce Eckel. Thinking in Java, 2002. Prentice Hall PTR, USA

[8] Calvin Austin, Monica Pawlan. Java2 advance programming. Mechanical Industry Press, 2001.

[9] Sun wireless development website: http://wireless.Java.sun.com/.

[10] Zhuang Yi. Wireless application development based on J2ME framework, ISSN: 1006-2475, 2002.

[11] , http://www.joyamigo.com/cjm/, 05-11-2000.

[12] http://www.cn-java.com,11-03-2001.

[13] Demestichas P, Vivier G,Khazen K,et al. Evolution in wireless systems management concepts: from composite radio to reconfigurability. IEEE Communications Magazine,2004, 42(5):90-98.